4.14 Analysis

This guide explains what to include in the Analysis section of the AQA A-level Computer Science NEA. The Analysis section is worth 9 marks and should show that you understand the problem, have researched it properly, have spoken to relevant users or stakeholders, and have produced clear objectives that define the scope of your project.

Purpose of the Analysis Section

The Analysis section should prove that the project is solving a real problem or investigating a meaningful area. It is not enough to say what you want to build; you must explain why the project is needed, who it is for, what problems currently exist, and how your proposed solution will address them.

Recommended Structure

1. Problem identification
2. Background research
3. Stakeholder or end-user research
4. Analysis of existing systems
5. Modelling of the problem
6. Requirements and constraints
7. Clear, measurable objectives
8. Success criteria

1. Problem Identification

Start by clearly describing the problem you are trying to solve. Avoid vague statements such as “I want to make a game” or “I want to create a database”. Instead, explain the real-world issue, the people affected by it, and why a programmed solution would be useful.

Good problem statement example: A small tutoring business currently tracks lesson bookings, student progress, and payments using separate spreadsheets. This causes duplicated data, missed updates, and difficulty producing reports. A single software system could reduce errors, improve organisation, and make it easier for tutors to manage students.

2. Background Research

Use research to show that you understand the problem area. This may include websites, books, articles, documentation, existing software, interviews, questionnaires, observations, or small prototypes. Do not include research simply to fill space; explain how each source helped you make decisions about the project.

- What did you learn from the research?
- Which features are common in similar systems?
- What limitations or gaps did you find?
- How did your research influence your objectives?

3. Stakeholder or End-User Research

It is useful to involve a third party, such as a potential user, teacher, client, expert, or someone familiar with the problem area. Their input helps show that your objectives are based on real needs rather than assumptions. You could use interviews, questionnaires, email conversations, meeting notes, or feedback on a prototype.

When writing this part, summarise the most important points from the stakeholder. Include evidence in an appendix if needed, but make sure the main Analysis section explains how the feedback shaped your project.

4. Analysis of Existing Systems

Compare your planned system with existing solutions. This could include commercial software, websites, apps, spreadsheets, manual processes, or previous systems used by the stakeholder. Focus on strengths, weaknesses, and features that are relevant to your own project.

Existing system	Useful features	Limitations	Impact on my project
Example booking spreadsheet	Stores student names, lesson dates, and payment status.	Hard to validate data, no automated reminders, and reports are manual.	My system should include validation, automated searches, and simple reporting.

5. Modelling the Problem

Include models that help you understand the problem and prepare for the design stage. The exact models depend on your project, but they should be relevant and useful rather than decorative.

- Data modelling: entities, attributes, relationships, sample records, or database requirements.
- Process modelling: flowcharts, data flow diagrams, state diagrams, or user journeys.
- Mathematical modelling: formulae, scoring systems, algorithms, or simulations.
- Interface modelling: sketches of how users will interact with the system.

6. Requirements and Constraints

Explain what the system must do and what limits apply to the project. Requirements describe the expected behaviour of the system, while constraints describe restrictions such as time, hardware, software, user ability, data protection, or technical limitations.

- **Functional requirements:** what the system must do, such as add records, search data, calculate results, authenticate users, or generate reports.
- **Non-functional requirements:** qualities the system should have, such as usability, reliability, speed, security, maintainability, and accessibility.
- **Constraints:** limits affecting the project, such as development time, available libraries, device compatibility, or the amount of real data available.

7. Objectives

Your objectives are one of the most important parts of the Analysis section. They define exactly what the project will achieve and will later be used in the Testing and Evaluation sections. Objectives should be specific, measurable, realistic, and linked to your research.

Weak objective: The system will be easy to use.

Improved objective: The system will allow a tutor to add, edit, delete, and search student records using a graphical interface, with validation to prevent blank names, invalid email addresses, and duplicate student IDs.

8. Success Criteria

Success criteria explain how you will know whether each objective has been achieved. They should be testable. For example, if an objective says the system will calculate a grade, the success criterion should state the expected inputs, processing, and output that will prove the calculation works correctly.

Objective	Success criterion
The system will allow users to search for a student by surname.	When a valid surname is entered, the system returns all matching student records within three seconds and displays the student name, contact details, and current status.
The system will calculate the total amount owed by a student.	Given a set of completed lessons and payments, the system correctly calculates outstanding balance using the agreed hourly rate and payment history.

Checklist for a Strong Analysis Section

- The problem is clearly explained and justified.
- The intended users or stakeholders are identified.
- Research is relevant and used to make project decisions.
- Existing systems are compared meaningfully.
- Stakeholder feedback influences requirements or objectives.
- Models are included where they help explain the problem.
- Requirements include both functional and non-functional points.
- Objectives are numbered, specific, measurable, and realistic.
- Success criteria can be used later for testing and evaluation.
- The scope of the project is clear and not too broad.

Common Mistakes to Avoid

- Writing a project idea without explaining the real problem.
- Including research without analysing what it means.
- Using objectives that are vague, subjective, or impossible to test.
- Changing objectives later without a clear reason.
- Ignoring the views of users or stakeholders.
- Making the project too simple, too broad, or unrelated to A-level Computer Science skills.
- Leaving out evidence that the problem has been properly investigated.