

3.1 Front end, back end, and security

Name: _____ Class: _____ Date: _____

Total: 15 marks

KEY WORDS front end • back end • request • response

Objective

Explain the front-end/back-end request-response model and protect secrets and enforced rules.

You must be able to:

- distinguish front end from back end
- describe a request and response
- decide which information and rules must stay on the server

1 Key knowledge and worked examples

■ Keep secrets off the page

The **front end** runs in the visitor's browser. The **back end** runs on a server controlled by the organisation. A browser sends a **request**; the server returns a **response**.

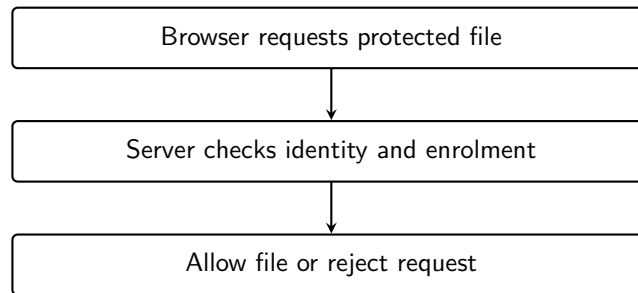
An API key, password, and database connection belong on the back end. Browser code is visible to every visitor. A rule such as “only enrolled students may download this file” must be enforced on the server.

■ Key knowledge: Trace an access decision

A hidden browser button does not enforce access. The server checks the identity and permission on every protected request before returning the file.

■ Worked example: Trace an access decision

An enrolled student requests `/files/lesson.pdf`. The browser sends the request and its session information. The server checks the signed-in identity and enrolment, then returns the file if allowed. Another visitor manually types the same URL; the server rejects that request because the permission check fails. The rule remains effective even when the visitor changes browser code.



2 Practice

2.1 State where front-end code runs. [1]

2.2 State what a server returns after receiving a request. [1]

2.3 Explain why an API key must not be placed in browser JavaScript. [2]

3 Exam-style questions

3.1 A course site checks in JavaScript whether a user is enrolled before showing a PDF link. Explain why this is not enough to protect the PDF. [4]

3.2 Recommend a secure request-response process for downloading the PDF. [4]

3.3 A website hides a download button for non-members but serves the file to anyone who types its URL. Identify the weakness and describe the server-side correction. [3]
