

3.1 Submission specification

Table of Contents

- [Report](#)
 - [Report structure](#)
 - [Report sections](#)
 - [Application Overview](#)
 - [Application architecture](#)
 - [Architecture diagram and overview](#)
 - [Justification of architecture](#)
 - [Response to marking criteria for project](#)
 - [Estimate the cost of your solution](#)
 - [Scaling up the application](#)
 - [Securing the application](#)
 - [Sustainability](#)
 - [Report quality](#)
- [Video submission requirements](#)
 - [Video structure:](#)
 - [Project Core - Microservices + Additional microservices + Serverless + ECS](#)
 - [Project Core - Load distribution](#)
 - [Project Additional - Dead letter queues](#)
 - [Project Additional - Communication mechanisms](#)
 - [Project Core - Auto scaling and custom scaling metric](#)
 - [Project Core - HTTPS](#)
 - [Project Additional - Container orchestration features](#)
 - [Project Additional - Infrastructure as Code](#)
 - [Project Additional - Edge Caching](#)
 - [Project Additional - Upon request](#)
- [Code submission requirements](#)

Report

- Your report must be submitted to [CAB432 A03: Cloud project \(Report\)](#) (<https://canvas.qut.edu.au/courses/20367/assignments/205185>)
- The submission format must be PDF
- Your report must use the template (choose any of the following formats):
 - Markdown: [CAB432_A3_Report_Template.md](#) (<https://canvas.qut.edu.au/courses/20367/files/7024493/download>)
 - LaTeX: [CAB432_A3_Report_Template.tex](#) (<https://canvas.qut.edu.au/courses/20367/files/7024494/download>)
 - DOCX: [CAB432_A3_Report_Template.docx](#) (<https://canvas.qut.edu.au/courses/20367/files/7024495/download>)
 - Pandoc can convert from Markdown to many other formats if none of the above are suitable.

Report structure

- Application Overview
- Application Architecture
 - Architecture diagram
 - Justification of architecture
 - Response to criteria for project implementation
- Cost estimate

- Scaling up
- Security
- Sustainability

Report sections

Application Overview

Give a brief (1 paragraph) overview of what your application does.

Application architecture

Architecture diagram and overview

Your diagram should show all services that you use and how they connect to each other. It should be made digitally and be easily readable within your report.

- [CloudCraft](https://www.cloudcraft.co/) [↗](https://www.cloudcraft.co/) has a diagram editor that can be used for free
- [draw.io](https://www.drawio.com/) [↗](https://www.drawio.com/) has an AWS shapes set (use one of the AWS templates or click *+More Shapes* and add the AWS shapes sets.) See [draw.io blog post](https://www.drawio.com/blog/aws-diagrams) [↗](https://www.drawio.com/blog/aws-diagrams) for more info.
- [Drawing and diagramming tools](https://aws.amazon.com/architecture/icons/) [↗](https://aws.amazon.com/architecture/icons/) lists several other options.

Briefly discuss the main use of each service. This part can be in bullet points.

Justification of architecture

Discuss your reasoning for the major choices in the cloud architecture. Your discussion should cover:

- Choice of division into microservices (i.e. why these microservices instead of some other way that you could divide up functionality)
- Choice(s) of compute (EC2 vs. ECS vs. Lambda)
- Choice of communication mechanisms and load distribution mechanism
- Choice of using service abstractions (API gateway, load balancer) or not

Your justification should make reference to (where appropriate):

- Microservice resource requirements, and in particular the requirements of your CPU intensive process
- Application functionality

You can also make reference to:

- Ease of implementation
- Scalability

Please note that this section is about *why* your architecture is as it is; it is not a description of the architecture.

Note that several "Project additional" criteria ask for a rationale, which you will include in the next section.

Response to marking criteria for project

This section is analogous to the response to marking criteria document from earlier assessments. There is a separate section for each criterion. The information required is indicated in the template.


Estimate the cost of your solution

Although cost optimisation is not the primary focus of this project, this criterion involves estimating the monthly cost of your solution, assuming an average demand of 50 concurrent users.

You will need to estimate what 50 concurrent users looks like from the perspective of resources utilisation. For this you can do some experimentation by using the application yourself for a while (say, 15 minutes) and see what impact this has on metrics like average CPU utilisation, RAM usage, bucket storage, and so on. AWS has a lot of metrics that you can access through CloudWatch or the monitoring tab on the details page for many services. It's not important that this estimate be super accurate, but you should try to get the order of magnitude.

If your application won't currently support 50 concurrent users, choose smaller number and specify this in your report.

You should use the AWS Pricing Calculator.

- <https://calculator.aws/>  (<https://calculator.aws/#/>)
- You will need to provide the public share link that is generated by clicking "Share" in the summary of your estimate.

In your report, provide a summary which includes the total price for each AWS service, along with the public link.

If you wish to provide a cost breakdown from a different tool, please ask the teaching team.

Scaling up the application

Suppose that your application is going to be deployed to support up to 10000 concurrent users. Discuss what changes would be required to your application in order to support this number of users.

Your discussion should cover:

- Microservices: would you change the current arrangement? How and why?
- Compute: would you use other types of compute? Are changes to auto-scaling required?
- Load distribution: would you need additional types of load distribution or a change in the mechanism?

Justify your proposed changes.

Securing the application

Suppose that your application will be used commercially and that successful cyber attacks against your application would drive away customers, costing you a lot of money. Discuss the measures that you have already taken, or could take, to secure your application. For each measure, mention a security principle that it relates to (eg. least privilege, separation of duties).

It is not necessary to discuss security controls that would be implemented outside of the cloud environment, such as educating staff, pen-testing, etc.. You only need to discuss controls that you would implement within your application and its cloud environment.

Sustainability

What choices can you make to improve the sustainability of your application. Consider how your choices can make an impact at each of these levels:

- Software
- Hardware
- Data centre
- Resources

Depending on your application you may not have much to say at some levels, in which case you can discuss some choices that you *could* make supposing that your application had additional relevant functionality where those choices would make sense.

Justify your choices.

Report quality

We will also evaluate the overall quality of your report in terms of writing style, understandability for an audience of your peers (i.e. would another CAB432 student be able to understand your report), formatting, presentation, appropriate use of citations where applicable, etc..

If you do cite external resources, you can use any of the citation styles given at [QUT cite|write](https://www.citewrite.qut.edu.au/) (<https://www.citewrite.qut.edu.au/>).

Video submission requirements

Important! Ensure that your video is actually submitted. You need to click the *Submit* button after uploading your video.

- Your video must be submitted to [CAB432 A03: Cloud project \(Video\)](https://canvas.qut.edu.au/courses/20367/assignments/202269) (<https://canvas.qut.edu.au/courses/20367/assignments/202269>)
- Only one partner needs to submit.
- Your video must be no longer than 10 minutes
- Your video must follow the structure below and demonstrate your application's functionality as requested
- Your video must be a screen capture at a high enough quality that the marker can make out details such as text. A recording from your phone is not acceptable.
- The purpose of the video is to *demonstrate* functionality, not explain it. The response to criteria document is used for justifying choices. Keep things brief in the video; you only need to describe what you are demonstrating.

Creating your video:

- You can record short videos separately and edit them together
- If some operation takes a long time (eg. uploading a video) you can edit it down to save time
- [OBS Studio](https://obsproject.com/) (↗ <https://obsproject.com/>) is a reasonable choice for screen capture, but you can use other software if you like.
- If you are working with a partner, you can record the video together over Zoom and use the recording for your submission
- Canvas has a built in option for screen capture which you can access through the submission page. Choose the *Record media* option.

Video structure:

Project Core - Microservices + Additional microservices + Serverless + ECS

Name each microservice in your application and show how it is deployed on AWS. For example, show the EC2 instance, Lambda function, or ECS Task/Service that is running it. *Briefly* mention what each service does (eg. "This service grabs videos from S3, transcodes them, and uploads the result to S3.")

Project Core - Load distribution

On the AWS console, show the mechanism that you are using for load distribution (eg. ALB, SQS queue). Also mention and show related cloud resources such as target groups.

Project Additional - Dead letter queues

On the AWS console, show messages that fail to process appearing in the relevant DLQs. Also mention what happens to the messages in the DLQs.

Project Additional - Communication mechanisms

On the AWS console, show other services that your application uses for communication (eg. API gateway, EventBridge, etc.) other than the load balancing mechanism. Briefly discuss what services use them and what for (eg. "the API gateway routes client requests for the static front end files, API, and progress reporting to the appropriate microservice.")

Project Core - Auto scaling and custom scaling metric

On the AWS console, show the auto-scaling group or ECS Service you use for automatic scaling. Show the scaling policy in use (eg. Target Tracking Policy). If using a custom metric, show how it is implemented (eg. CloudWatch metric, Lambda, etc. as appropriate to your implementation).

Demonstrate your application automatically scaling out from 1 to 3 instances and back down to 1 instance. You do not need to show this in real time, as it will take several minutes for this to happen completely. Instead, edit out the waiting and just show what is happening at these moments:

- Initial state with no load, and starting the load
- Third instance added and load stopped
- Back to one instance

Show graphs in the AWS console at each of these moments, showing the In Service Instances count and the relevant scaling metric (eg. Average CPU utilisation). For EC2 these can be accessed from the Monitoring tab of your Auto Scaling group. For ECS, show the Health and Metrics tab which shows CPU utilisation and the number of tasks, and the Events tab showing tasks starting and stopping. Alternatively, you can create a custom CloudWatch dashboard including the appropriate graphs. The graphs and/or event logs should show enough information to verify that the number of instances has gone from 1 to 3 and back to 1, following the target metric.

Project Core - HTTPS

Show your application being accessed in your browser using HTTPS with your custom domain name. The lock icon and `https://` should be visible in the address bar of your browser. In ACM show the certificate in use. Show the API Gateway or ALB configuration using the certificate.

Project Additional - Container orchestration features

In the AWS console, show the configuration(s) relevant to the features that you are using. Demonstrate each functionality working. If the process takes a long time to complete (eg. rolling update) you can edit out the waiting similar to auto-scaling above.

Project Additional - Infrastructure as Code

It is not necessary to show an actual deployment. You can show the current state of your deployment, eg. with `terraform state list`, or showing your deployed Stack in CloudFormation on the AWS console.

Project Additional - Edge Caching

In the AWS console, show the CloudFront distribution you are using and the Origin configuration. Show the AWS resources being used for the Origin (eg. S3 bucket). If the Origin is a service in your application, briefly discuss what is being cached from the origin.

Project Additional - Upon request

Discuss with the unit coordinator about what you need to demonstrate for this criterion.

Code submission requirements

- Your code must be submitted to [CAB432 A03: Cloud project \(Code\)](https://canvas.qut.edu.au/courses/20367/assignments/205186) (<https://canvas.qut.edu.au/courses/20367/assignments/205186>)
- Your submission should not include `.git`, `node_modules`, `venv`, `__MACOSX` or other files that are not required for building and running your application (eg. video files left over from testing)

These requirements will make it easier for the markers to navigate your source code and find relevant information.

- If you have trouble submitting your code, please ensure that it is no more than 100Mb in size and that you haven't included any extraneous files or directories. (such as those mentioned above.)