Software Engineer - Interview Flow

V2025.2

marloesconsultinggroup.co.nz

INTRODUCTION

Despite both technology and the workforce becoming more diverse than ever in the 21st century, our interview processes have remained firmly lodged in the past.

The days of having a word document filled to the brim of questions from a decade ago are very much still here. Worse, the forced memorization of HackerRank and Leetcode type programming questions only ensures that an engineer can do just that, memorize very specific programming sets not often seen in a business context.

At Marloes Consulting Group, we take a different approach. We allow a candidate to express themselves in a safe environment, letting them “brag” about their best accomplishments while also getting a sense of where the candidate is strong or weak.

The key to a good interview is not forcing them into a box or trying to trip them up, but instead allowing a candidate to show you who they really are.

Wade Gausden – Managing Director

KEY INSIGHTS
BAD INTERVIEWS MATTER

Employers tend to underestimate how bad interviews and application processes can negatively affect not only candidates they have in the pipeline, but those candidates can then discourage others in the marketplace from applying.

In a market as small as New Zealand, a negative experience and quickly make its way through word of mouth, stalling out recruitment drives.

49%

Turned down an offer due to a bad recruiting experience

56%

Say they would discourage others from applying due to bad recruiting

https://www.pwc.com/us/en/services/consulting/business-transformation/library/hr-recruiting.html

KEY INSIGHTS
CODING TESTS DON'T WORK

Programming tests such as Leetcode, or take home tests don't work for the simple reason they are self selecting. These tests are a time check.

Does the candidate have enough free time to memorize Leetcode problems?

Does the candidate have no other time commitments and can dedicate several hours to a take home test?

When a large portion of your candidate base is already employed, you filter these people out.

51%

Workers who currently have a job are either actively seeking, or open to a new job

https://www.gallup.com/467702/indicator-employee-retention-attraction.aspx

KEY INSIGHTS
LUCKY EXPOSURE

Software engineers often fall into the trap we call “Lucky Exposure”. That is, a software engineer can only work on (commercially) what their place of work provides for them.

If their organization has a large legacy monolith written in VB.NET, that doesn’t make them a bad engineer, just unlucky.

The interview flow in this guide attempts to overcome these sorts of biases by allowing an engineer to talk technology in terms and examples that make sense to them, not force them into gotchas and traps.

58%

Engineers who say they have little to no influence on technology purchases

45%

Engineers who say knowledge silos stop me from getting ideas across the org

https://survey.stackoverflow.co/2024

KEY INSIGHTS
STRUCTURE REMOVES BIAS

The interview flow described in this guide might have one (or several) steps that seem obvious. However, it's important that the same interview is given to each candidate for a given role.

Studies tell us that non-structured interviews lead to bias. But common sense will tell us that by giving a standardized set of questions to every candidate means that every person has the same chance to show off the same skills.

The interview flow described in this guide is more akin to a "semi structured" interview. The same questions are presented to a candidate, but because of the diversity in possible answers, the interviewer must be experienced enough to rate the responses how they see fit.

https://www.researchgate.net/publication/308753199_Do_structured_interviews_eliminate_bias_A_meta-analytic_comparison_of_structured_and_unstructured_interviews

"Unstructured interviews were significantly more susceptible to bias than were structured interviews."

INTERVIEW FLOW USING THIS GUIDE

The following pages outline an interview process we use at Marloes Consulting Group to assess someone's technical skills as a software engineer.

The process is more of an outline, and does not follow a strict "question with a right answer" type flow, instead it allows a candidate to express what they've worked on, including allowing them to "brag" (In fact it's encouraged!).

The interviewer should be experienced enough in technology to understand that a candidate's answers are never necessarily wrong, and instead simply nudge the candidate to talk through their experience.

Key questions

On the right hand side of every interview flow, a "key questions" section will be shown on the right. These are the questions should be able to answer given a candidates responses. These are questions that **you the interviewer** must be able to answer, not the candidate!

While you should let the candidate speak and express themselves, you at times may be required to "drive" the interview in a direction to get answers to these key questions.

INTERVIEW FLOW
INTRODUCTION

Rationale

The introduction section of the interview flow is simply a way to kick off the interview in an easy going manner.

The core piece is to let the candidate describe what they've been working on without prompting. Instead of asking "So I see you've been working with Azure, Do you use Azure SQL?", let the candidate express themselves how they want to be seen.

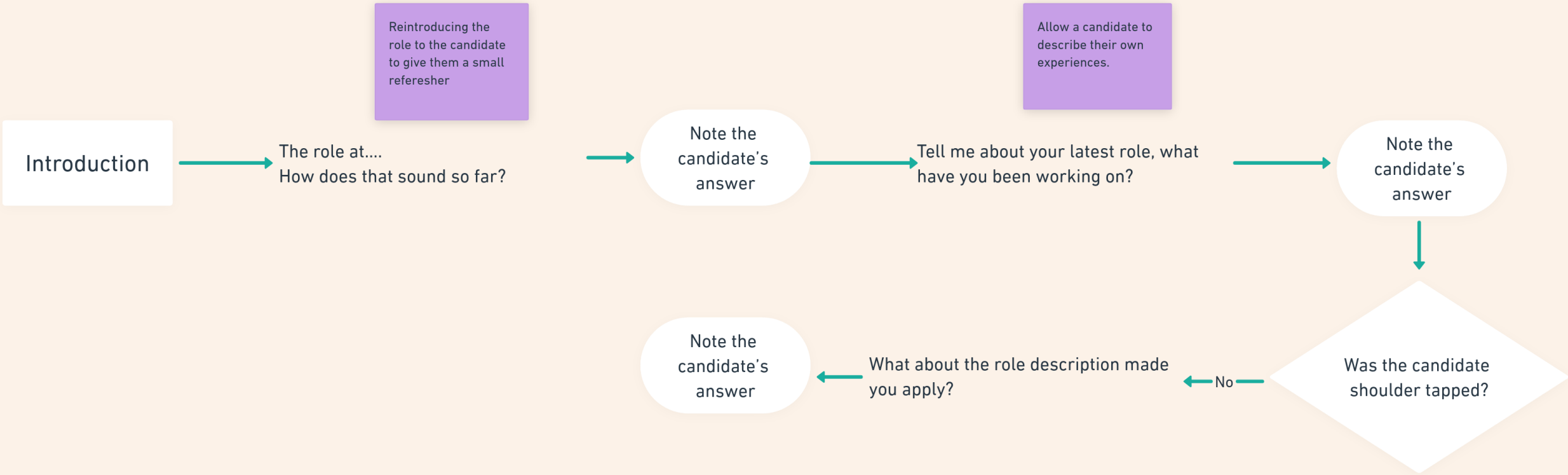
Key questions

What has the candidate been working on?

How does the candidate describe themselves? E.x. Do they jump immediately to devops practices, to backend development, to cloud engineering. This is important to understand how the candidate views themselves.

If the candidate applied, what drew them to apply? ("I need a job" is *always* a satisfactory answer)

INTERVIEW FLOW
INTRODUCTION



INTERVIEW FLOW
TECHNOLOGY CHOICES

Rationale

The technology choices section of the interview flow is designed to allow a candidate to talk about the technology, programming languages, and frameworks they already work with. It gives the chance for a candidate to kick off the interview in an area they already know and love.

At any stage of an engineer's career, they can be siloed or pigeonholed. This is not a mark on anyone's ability, however they should be able to describe the application they work on in as much detail as possible.

Key questions

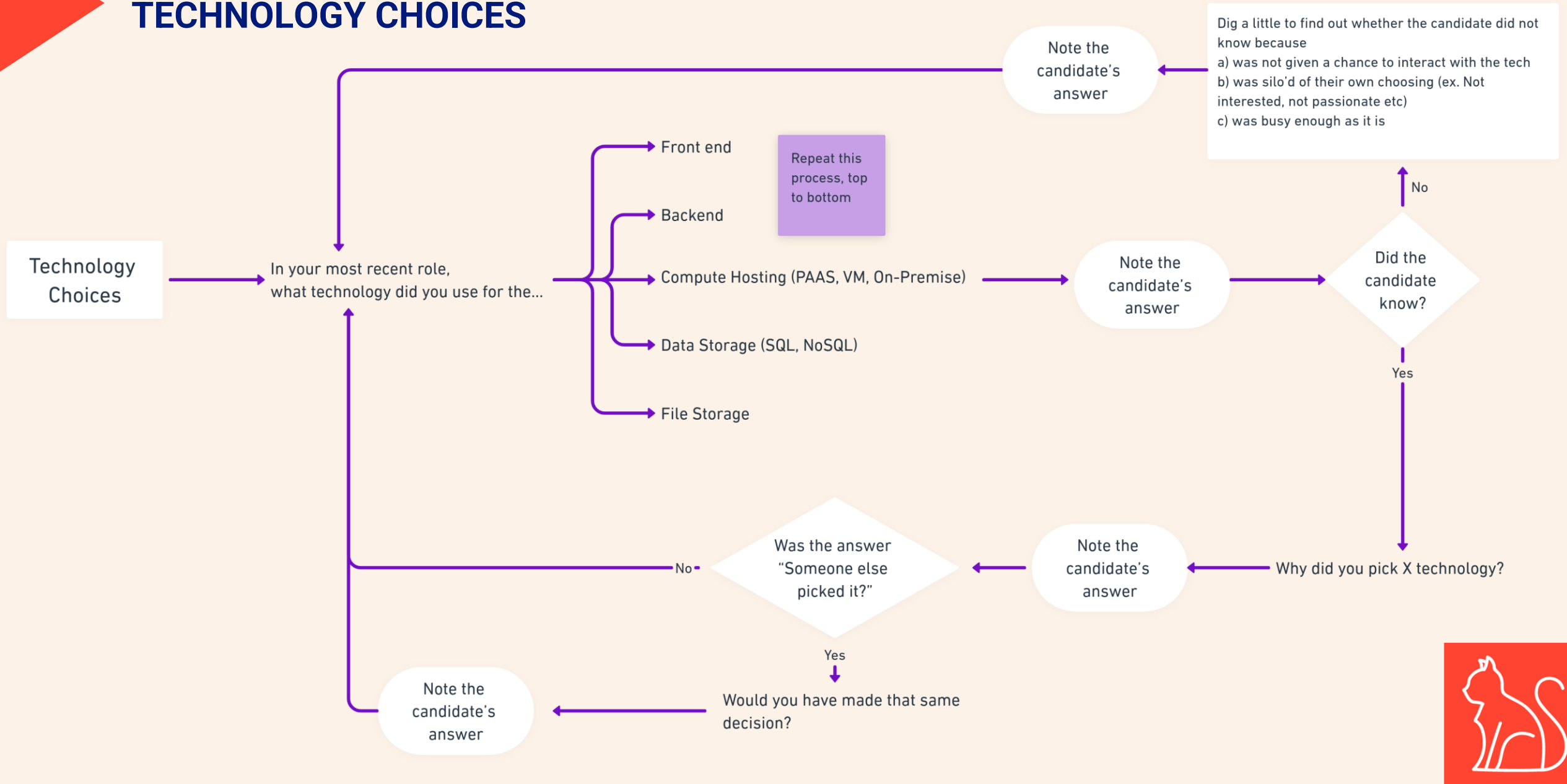
Has the candidate worked on what they claim in their CV?

Can they describe the technologies in play, even if it's not their area of expertise?

Can they rationalize technology choices, even if it was the right one at the time but wrong in hindsight?

Can they take ownership, responsibility, and accountability for parts of the application?

INTERVIEW FLOW
TECHNOLOGY CHOICES



INTERVIEW FLOW
APP ARCHITECTURE

Rationale

It is rare that an individual engineer will have the ability to decide or change an entire applications architecture within an organization.

However, it should not be rare that an individual engineer understands which architecture is currently in use, drawbacks of said architecture, and alternatives.

If an engineer is highly specialized (e.x. Mobile developer), still attempt this section driving them towards patterns in their specialist area (e.x. MVVM)

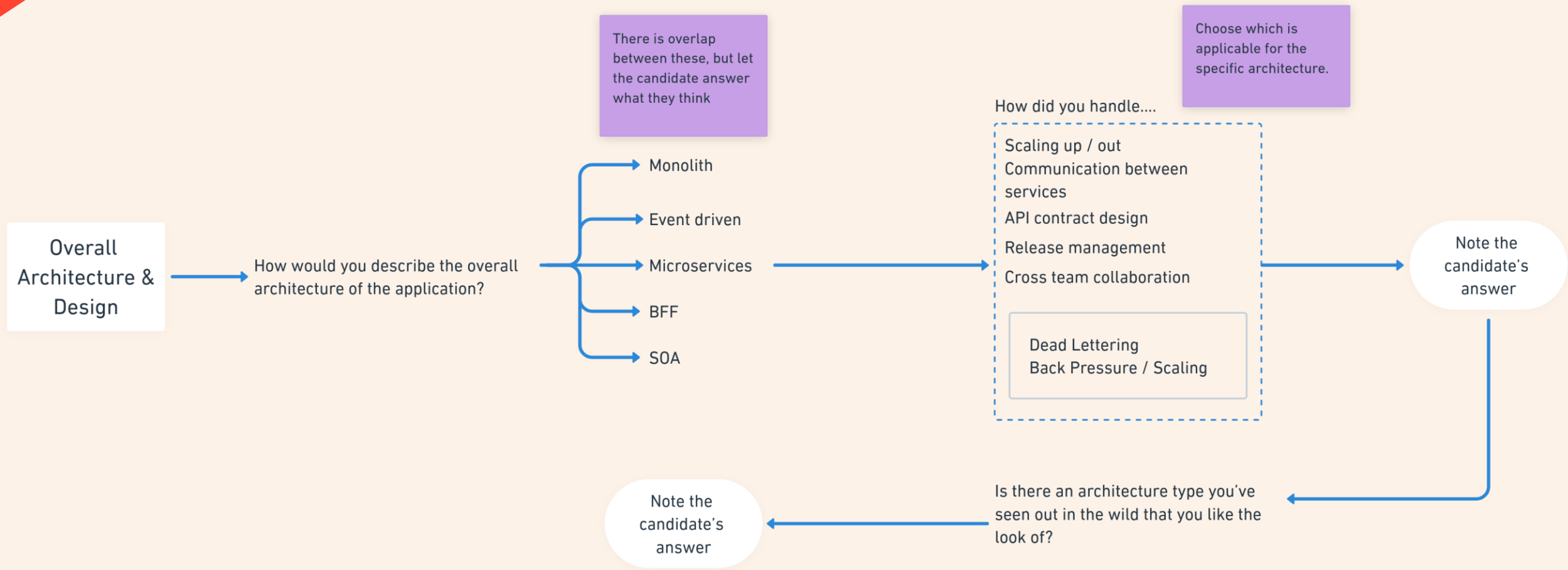
Key questions

What is the overall architecture that the candidate has worked with?

Can they describe concepts related to their specific architecture? Including any mistakes or pitfalls they found along the way?

Can they identify alternative popular architectures?

INTERVIEW FLOW
APP ARCHITECTURE



INTERVIEW FLOW PRACTICES

Rationale

Practices in this context refers to all the things in the periphery of a developers main job. Automated tests, CI/CD, logging etc.

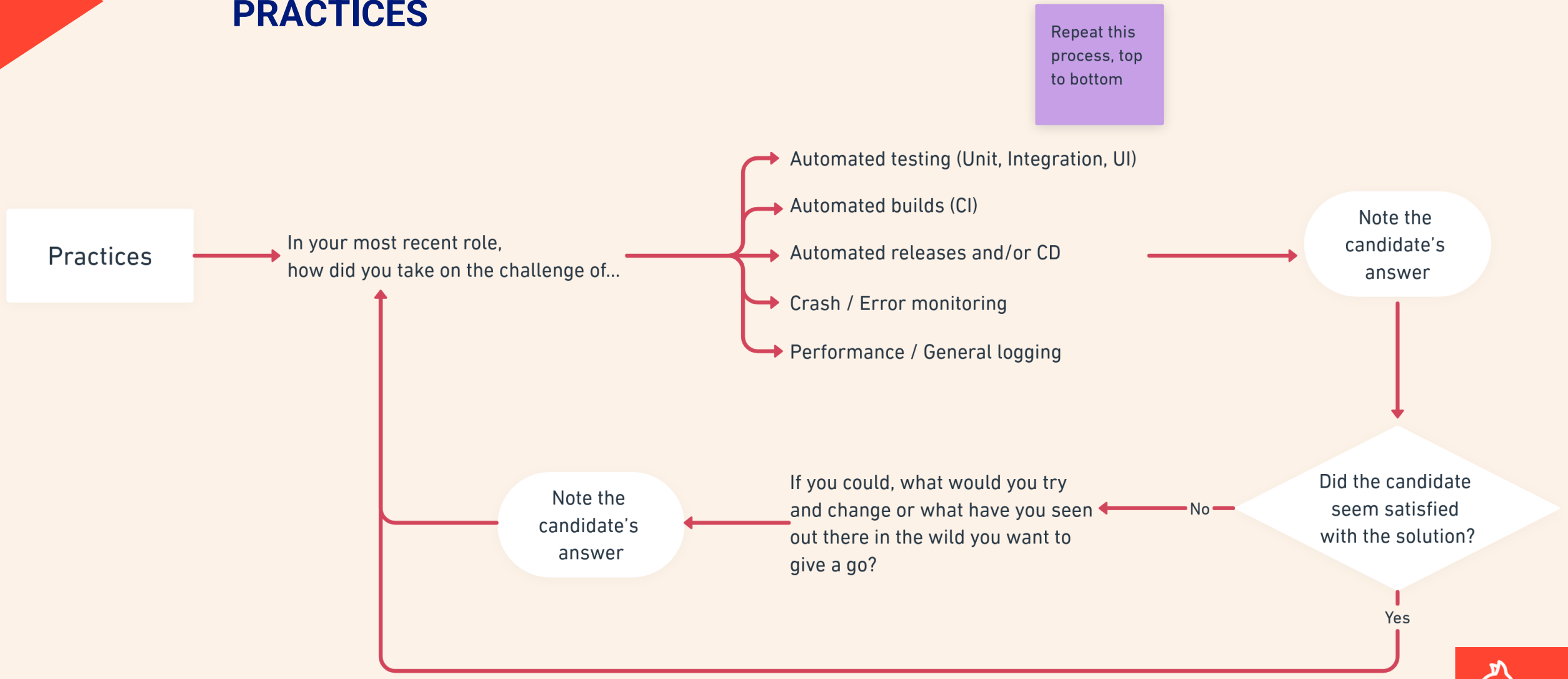
Again, there isn't necessarily a right or wrong answer, it's more about the candidates mindset. The important step is if a candidate sounds unsatisfied with what they are currently doing (e.x. No automated tests), is to give them the opportunity to say what they would do in an ideal world.

Key questions

Does a candidate have a good grasp on practices outside of their main developer role?

If they are lagging a little behind in modern practices, do they recognize that and have an idea on how they could do better?

INTERVIEW FLOW PRACTICES



INTERVIEW FLOW
ADHOC

Rationale

Some roles and/or organizations have very specific needs. This flow allows for 3- 5 questions that can be asked of a candidate.

However, the same questions must be presented to every candidate. Do not have a bank of questions and you pick a few each time as this introduces bias.

Examples of questions :

“Have you ever worked with hybrid networks between cloud and on-premise?”

“Have you used Storybook for front end development?”

Key questions

Try and come up with your key questions that you want to answer. These should be thought of ahead of time and not adhoc after each candidate’s response.

Again, try and not “trap” a candidate with a right or wrong answer.

INTERVIEW FLOW ADHOC

Ad Hoc

Choose 3 - 5 questions to ask a candidate. These must be asked to **every**candidate.

The questions should not necessarily have right or wrong answers, but give the candidate a chance to explain more about themselves.

Note the candidate's answer

INTERVIEW FLOW PROCESS

Rationale

Agile, Scrum, XP, Kanban, Jira, Azure Devops, Notion. At times all of these methodologies and task tracking systems blend into one and for the most part, there isn't a huge amount of difference between them.

However, a candidate that has experience utilizing Jira and Scrum, might just get up to speed a little faster than a developer who is used to getting tasks via an excel spreadsheet and working adhoc.

This section doesn't have any right or wrong answers, it's simply to gauge how a candidate has worked in the past.

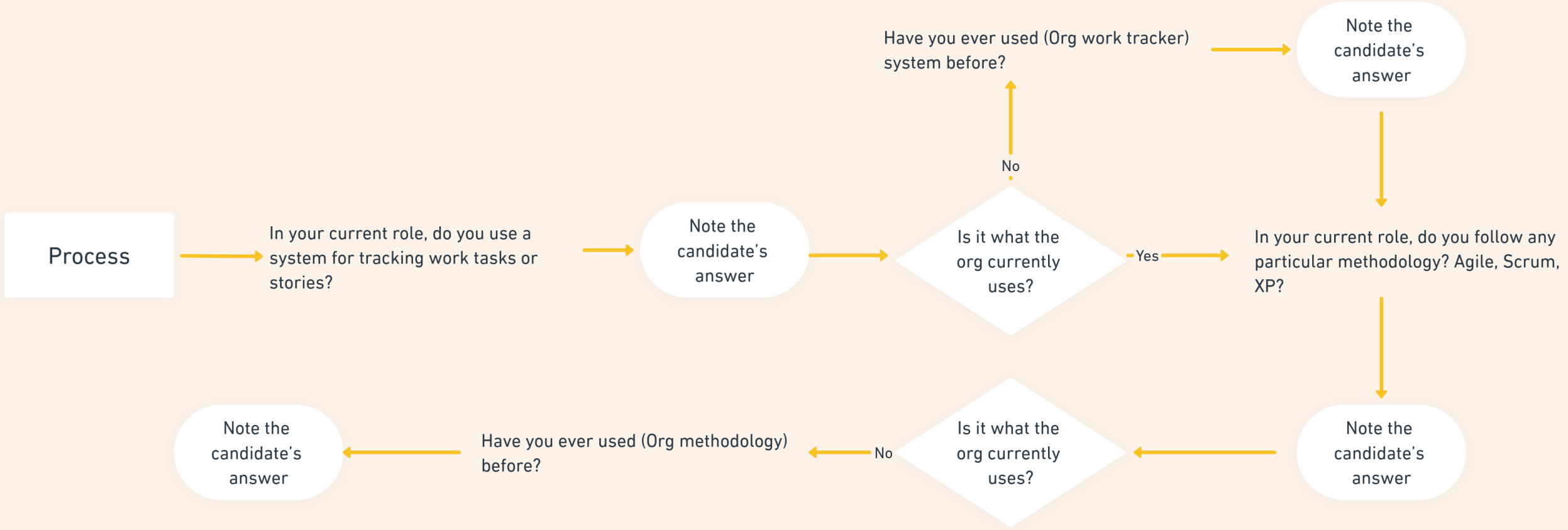
Key questions

What work task tracking systems has the candidate used before?

What methodologies/frameworks for developing software has the candidate used before?

Has the candidate utilized processes similar to the organization?

INTERVIEW FLOW PROCESS



INTERVIEW FLOW
WRAP UP

Rationale

The final wrap up includes the standard “any questions for me”. However, the important part of this flow is giving the candidate one last chance to describe something they’ve built or most proud of.

Again, allowing a candidate to express themselves and talk about what they love working on gives us much better insight into how they operate.

Key questions

Did the candidate have one final passion project to talk about? (Not required)

Did the candidate have any thoughtful questions about the role or organization (e.x. Team makeup, Existing devops practices etc)

INTERVIEW FLOW
WRAP UP

Wrap Up

Just incase we haven't talked about it yet, I want to give an opportunity for you to talk about your favourite feature, application, or piece of work you are most proud of.

Can be commercial, hobby, maybe not even software related.

Note the candidate's answer

Thank you for your time.
Are there any questions about the role, organization, or process that you have. We have plenty of time.

Note the candidate's answer

Contact

Not sure where to start? Call us for a no strings attached chat on how we might be able to help.

marloesconsultinggroup.co.nz
09 888 4188
contact@marloesconsultinggroup.co.nz