Rolsa Technologies Software Testing and Feedback report

Methods Used

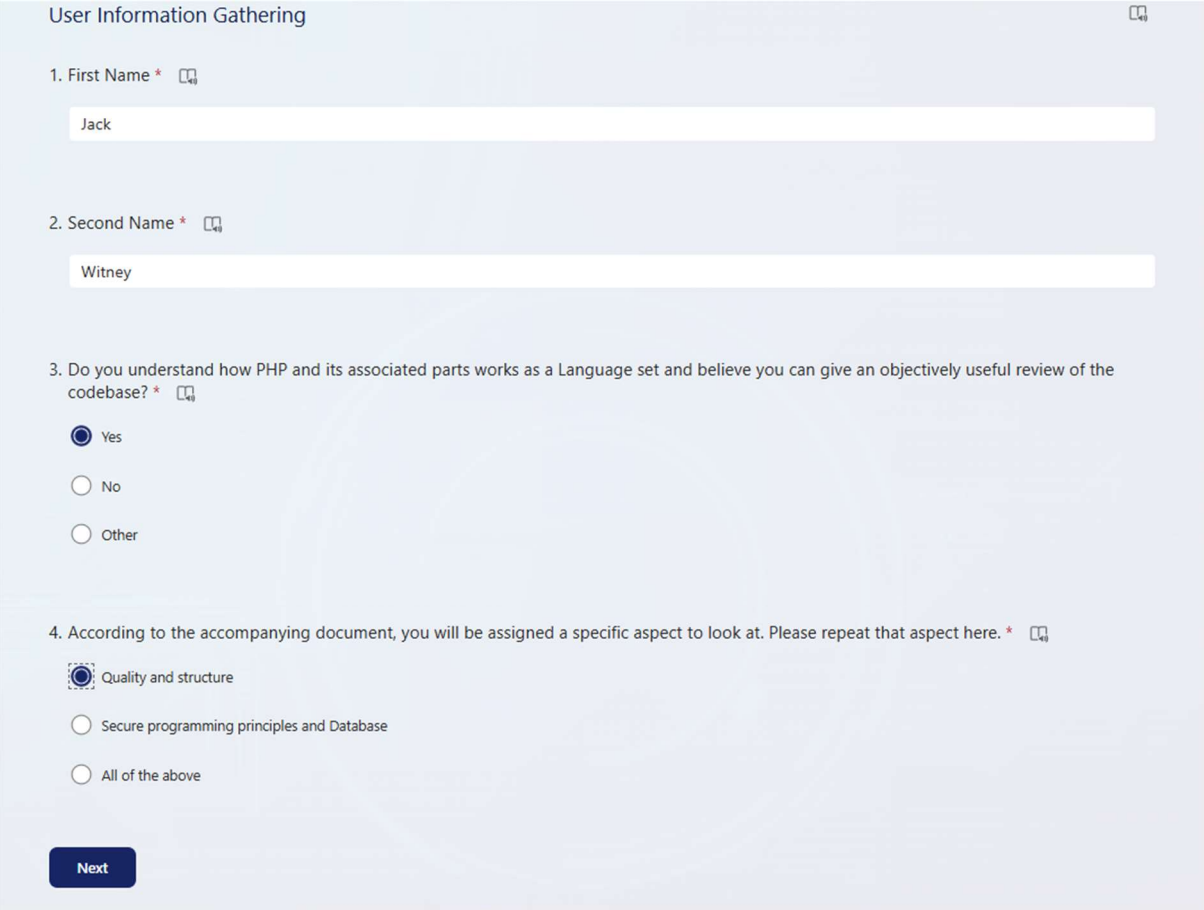
Digital Forms

The digital forms will include a rating out of 5 and an extended text box for contextualising the rating for later interpretations.

I will be building both technical and non-technical forms containing the above-mentioned methods, separating the non-technical forms (based on UI/UX) into different user groups. For this purpose, I have identified my available testers to be either “students” or “staff” and created a form to take feedback accordingly.

Technical Review Form

Design



First, I ask for the reviewer's first and second name. This helps us keep track of who submitted which feedback. It also means if there's ever any confusion or something needs to be clarified, we know who to reach out to

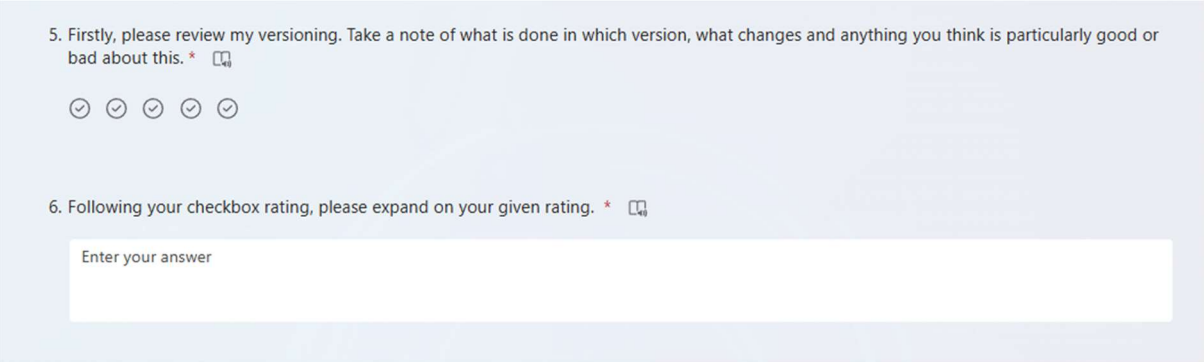
Next, I check if they understand PHP and its related tools well enough to review the codebase. It's a straightforward multiple-choice question: Yes, No, or Other. No need for a long explanation or a text box—just a quick check to make sure the person reviewing has enough familiarity to give useful input.

After that, I ask them to confirm which specific area of the codebase they have been assigned to review. This should be based on the accompanying document they received. The idea is that each person is randomly assigned one of the following areas:

- Quality and structure
- Secure programming principles and database
- All the above

This random assignment helps to create a more diverse data pool since and focus more reviews on the places most in need of review and feedback.

Code review for Quality and Structure

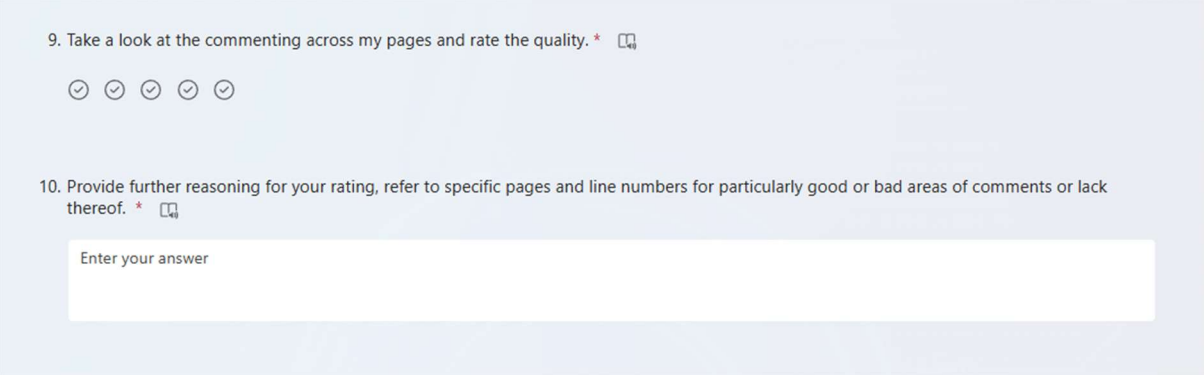


The real questions kick off with asking the user to review my use of iterative versioning, this is my first question because when the development environment is opened the first thing you are met with is the versioning. As such I thought it made good sense kick off with it.

I ask the user to review the versioning , what is done in each version , any changes between them in the way of refactoring and new pages etc as well as highlight anything they is particularly well done or poorly done. They are also asked to expand on their rating in a text box. This is a deliberate and consistent decision as understanding opinions in a technical sense from a rating is hard enough without having to try write up about it. You will see that each of my questions is accompanied with an extended box for any further points.

After the initial section, I ask them to take a proper look at the file directory structure of the project. Question 7 is a quick judgment call—do they think it follows common conventions or at least meets basic good practices? I give them a set of tick options to choose from, so it's simple and doesn't slow them down. It's meant to get a general sense of how clean or messy things look at first glance.

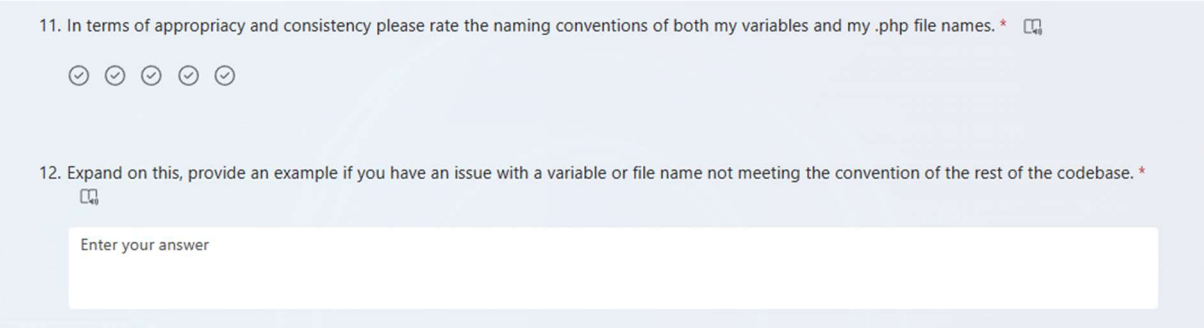
Then in question 8, I ask them to explain their reasoning a bit. If they thought the layout was great, I want to know why. If they didn't, I want to know what stood out as wrong. If folders are all over the place, or key parts of the structure are missing or confusing. This gives more depth to their rating and helps identify whether it's a minor issue or something more serious.



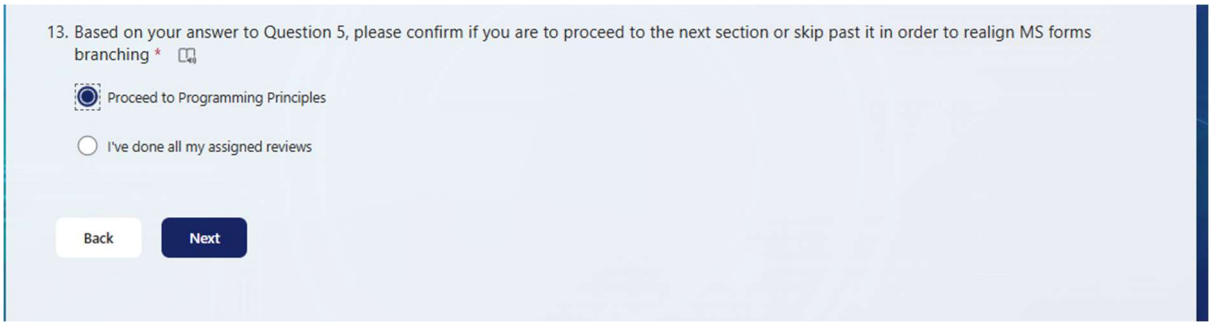
With Identical structure to the previous question set I now ask the reviewer to take note of the commenting quality and structure and rate it from 1 to 5. A standardised scale I believe is the most effective for this style of query. I toyed with the idea of providing a 1-10 scale however I discounted this idea as it can create a bit too much ambiguity. For example, a 7/10 is classed as a good mark on something like a maths test but for a code review leaves a lot of question up in the air about the legitimacy of the review. A 1-5 scale directly allows me to digest it very easily.

- 1 is appalling and not worth the bandwidth to host
- 2 is sub-par commenting much like what a gen AI provides
- 3 is still not considered a "neutral" and shows that while commenting is present it is not effective enough
- 4 improves on commenting present so perhaps it is frequent but not always useful or informative
- 5 is amazing commenting, clear, concise and effective breakdown of a script / code base

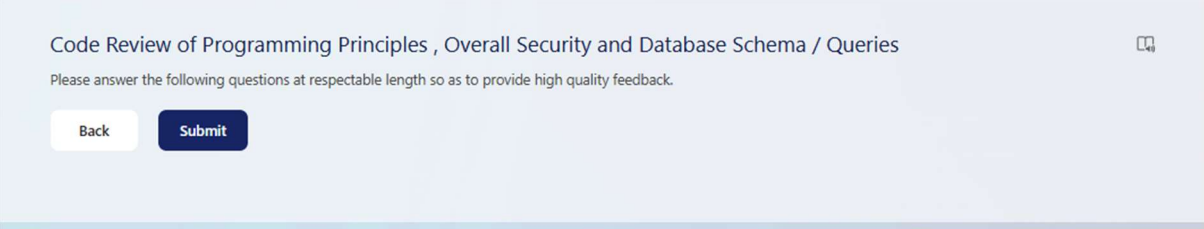
And now again I ask the user to provide a breakdown of their rating from the previous question. I ask them to specifically reference a page/ pages and line numbers where comments are particularly poor or very good. Making the user provide a page and line number I believe will help prevent any fraudulent ratings or comments.



Next up, I'm asking the user to review the appropriacy and consistency of my naming conventions. This question is miles better than asking the user to simply rate the variable names because then if I had one called "$fudge" users are likely to rate it highly because it's funny. Adding appropriacy into the equation helps me make sure the user is looking at what is important.



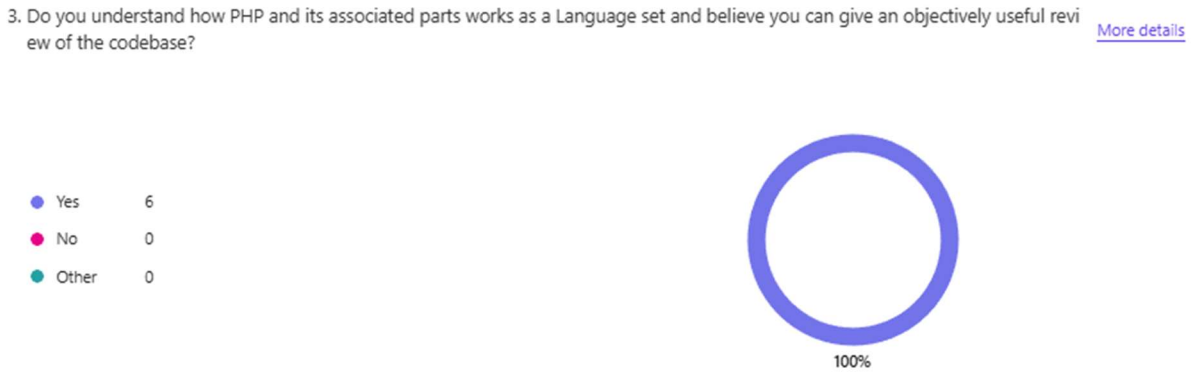
This question very simply is in place for the branching system in Microsoft forms. It doesn't contain any valuable information. If the user chose the done all assignments options, they will be taken to the end of the form. Otherwise, they are taken to start the reviewing process of the database queries and statements.



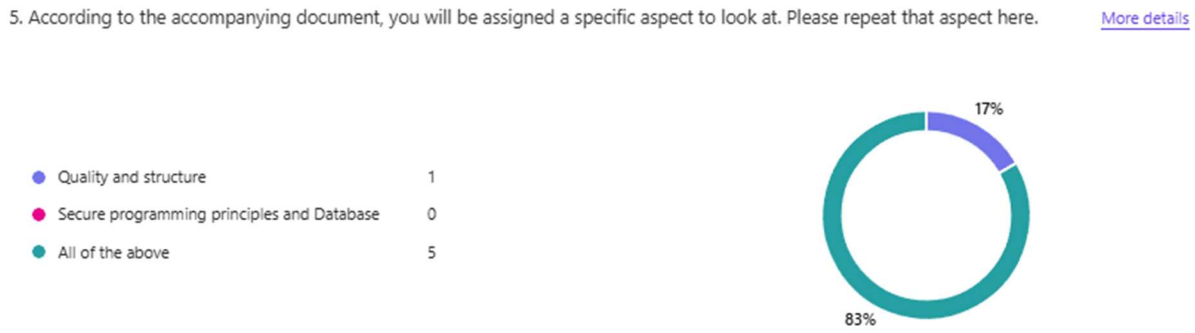
Unfortunately, due to the time I had scheduled to receive the feedback from my testers I wasn't able to properly build my feedback system for the security and database related questions.

As a compromise I was able to receive verbal feedback which I will discuss in due course.

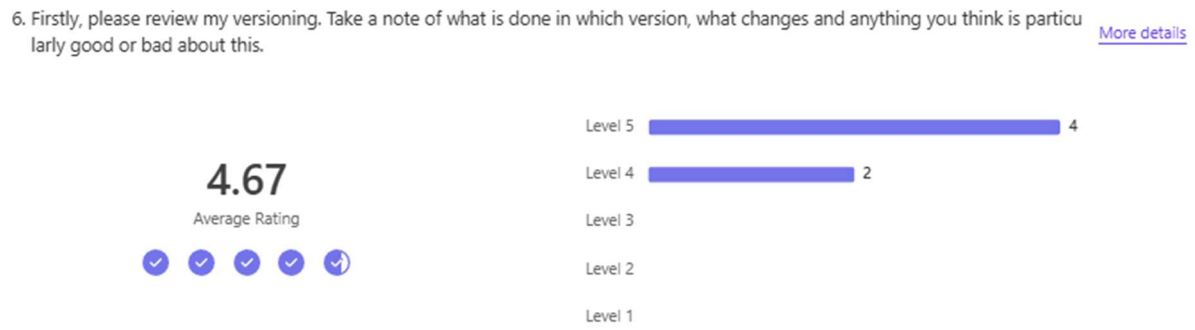
Results / Answers and Trends for Technical review



First question in, thankfully all 6 testers at this stage of review have checked "yes" to being knowledgeable in PHP and its associated parts which right off the bat ensures me that the technical feedback I receive will be well thought out and come from knowledgeable individuals.



Next up with where I've asked the users what part of the codebase, they will be reviewing it's predominantly a case of reviewing the whole codebase and not just a portion. I had anticipated for a larger audience of technical testers however unfortunately in the timeframe I had to gather the feedback I was only able to grab 6 good technical reviews



Now looking at the versioning reviews of my code I can clearly see an average score of 4.67 out of 5. This consists of:

- 4 lots of 5 stars
- 2 lots of 4 stars

Overall, I can say from the rating I am happy with the reviews, the versioning was clearly effective and well formed.

Looking now at specific comments we have:

"Versions are **clearly laid out** only side comment is that there could have been a notepad that with the small document to make it a little bit earlier."

“It's **broken down into 3 well visualised versions** and each of them **has clearly notable changes and further development** on the last”

“Great use of versions**. Conveys clear changes between versions & good progression routes** from one to the next.”

Extracting from these quotes it seems the consensus is that the versions are well thought through and provide clear and easily noticeable developments and changes from the previous version. The other 3 responses contain essentially identical responses in terms of key points and you can find them at “Task3_PartA_TechnicalDataAppendix_000018212_Witney_J”.

For file directory I’ve been scored an on the decimal 4.00. this consists of:

- 1 x 5 star
- 4 x 4 star
- 1 x 3 star

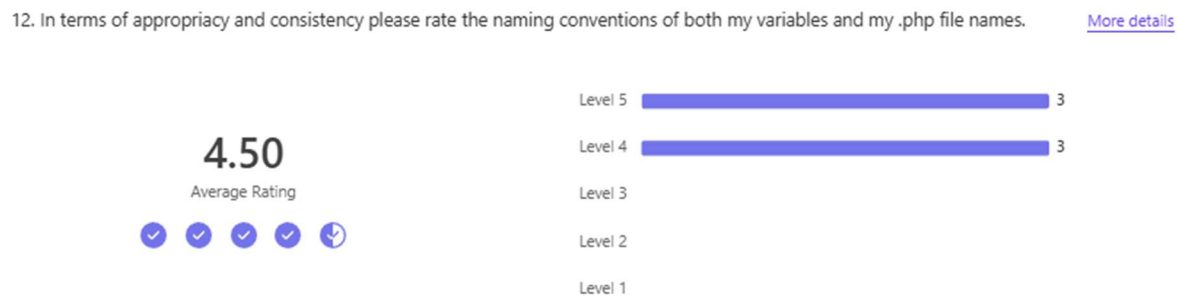
Taking a further written answer from the 4 star reviews I get:

"The chunks of code that were **admin are organized** into the admin folder. could have **maybe laid out folder based where the user based ones like login could have been structured into a folder for user** for example"

"**clear what is each folder, assets/documents could have had its own folder** so that it is clear what they are"

"**great layout overall** but **it does lack an index.php file in the root**. this will **stop the site from opening directly** into a page when heading to the host site (or in this case localhost)"

Reading through these quotes the same points recur. Seemingly the consensus here is that there could have been a bit more folder structure for stuff like the user pages grouped into a folder called user etc. One tester refers to the absence of an index page leading to the website not loading correctly when accessing the host domain.



For variable and page names I have been hit with a 4.5 out of 5 broken down into:

- 3 x 5 stars
- 3 x 4 stars

From the written comments section for this question the following 3 responses stuck out as the most helpful:

"**Appropriacy is an easy 5/5** they all make distinct sense and provide an accurate insight to the files contents however there are **some inconsistencies in the naming conventions**. Not massive issues just small quick fixes for version 4."

"**Appropriacy full marks** they are all spot on. The **issue lies with consistency where some are referenced slightly differently**. Easy fix so not a massive concern."

"**all** of the naming conventions are **consistent and clear** what they are being used for"

Unfortunately, despite having plans to build a feedback section regarding the security and database side of things I was sadly not able to create this in the timeframe before my planned review time. As a compromise, I did request some verbal feedback from the technical reviewers. On an average, my security and databasing was rated at about a 4/5 stars. Averaging this rating gives me a good insight that the code and principles were clearly MOSTLY there but lacking just in ways not implementable for the dev env.