### Teams Meeting Transcript

\*\*Date:\*\* 12 February 2024

\*\*Time:\*\* 10:31 AM - 11:00 AM

\*\*Attendees:\*\*

- Emma (Product Owner)

- Liam (Business Analyst)

- Zoe (Test Analyst)

- Mark (Solution Architect)

---

\*\*10:31 AM - Emma:\*\*

Good morning, everyone! Thanks for joining the call. Today, we are here to discuss the new feature for our streaming app: the ability to add, remove, and view items on a watchlist. Let’s start off by outlining the primary objectives of this feature.

\*\*10:32 AM - Liam:\*\*

Absolutely, Emma. The watchlist functionality will allow users to easily manage the content they are interested in. They should be able to add or remove items directly from the asset detail page to streamline the user experience.

\*\*10:33 AM - Zoe:\*\*

That sounds great. From a testing perspective, we need to outline the scenarios we’ll cover. We should verify that the user can add items to their watchlist, remove items, and view the list correctly. We also need to consider scenarios for handling items that are already in the list.

\*\*10:34 AM - Mark:\*\*

I agree, Zoe. In terms of architecture, we need to ensure our back-end services can handle the additional API requests for adding and removing items from the watchlist. I suggest we implement a RESTful API that can handle these operations efficiently.

\*\*10:35 AM - Emma:\*\*

Good point, Mark. Can you provide a brief overview of how the API might be structured?

\*\*10:36 AM - Mark:\*\*

Sure! We could have three main endpoints:

1. \*\*POST /watchlist\*\* for adding items.

2. \*\*DELETE /watchlist/{item\_id}\*\* for removing items.

3. \*\*GET /watchlist\*\* to retrieve the user's current watchlist. Each user will have their own watchlist stored in the database.

\*\*10:37 AM - Liam:\*\*

That structure seems solid. I’d like to discuss user permissions as well. How will we handle cases where users try to modify a watchlist that doesn't belong to them?

\*\*10:38 AM - Mark:\*\*

Good question, Liam. We need to ensure that every API call checks user authentication and authorization. Users should only be able to modify their own watchlists.

\*\*10:39 AM - Zoe:\*\*

I can create test cases around user permissions and scenarios for unauthorized access attempts. We also need to think about the UI/UX.

\*\*10:40 AM - Emma:\*\*

Yes, let's address the UI. The asset detail page should have clear buttons for adding and removing items, along with feedback messages. We want to make it intuitive.

\*\*10:41 AM - Liam:\*\*

How about a toast notification to confirm actions? Something like "Item added to your watchlist" or "Item removed from your watchlist."

\*\*10:42 AM - Zoe:\*\*

I like that idea, Liam. We should also verify that these notifications display correctly across different devices.

\*\*10:43 AM - Mark:\*\*

Additionally, I think caching the watchlist locally on the user’s device can improve performance, so they don’t have to make repeated API calls for fetching the list.

\*\*10:44 AM - Emma:\*\*

I love that approach, Mark. It will enhance the user experience. Let’s ensure all these points are documented and assign tasks accordingly.

\*\*10:45 AM - Zoe:\*\*

I will draft the testing strategy based on our discussions today. This will include UI testing for the notification messages and API testing for the watchlist operations.

\*\*10:46 AM - Liam:\*\*

I'll finalize the user stories and acceptance criteria based on these features.

\*\*10:47 AM - Mark:\*\*

I can begin working on the API design documentation, so we have a clear blueprint for implementation.

\*\*10:48 AM - Emma:\*\*

Great! Let’s aim to have a follow-up meeting next week to share progress and address any blockers. Does that work for everyone?

\*\*10:49 AM - All:\*\*

Yes, sounds good!

\*\*10:50 AM - Emma:\*\*

Awesome! Thanks, everyone, for your input. Let’s make this feature a success. Have a great day!

\*\*10:51 AM - Meeting Adjourned.\*\*

---

### End of Transcript