**Question 1:**

As a product owner in this Project, I suggest choosing the **AGILE** methodology, because:

**a. Requirements characteristics**

* + Reliability:
* The project requirements are relatively well-defined and comprehensive for the online fashion shopping project proposes an automated project for selecting, fitting, and purchasing women’s clothing.
* It can immediately run when the project finished.
  + Types and number of requirements:
    - The requirements are explicit and unambiguous.
    - They include more than 5 requirements with both functional and non-functional requirements.
  + How often the requirements can change:
    - Since this is the first time that human has faced to the pandemic that very huge negative impact like this so the requirements can change in the future.
    - The situation mentioned: “The organization had contracted with a local company to provide additional resources when needed.” So that the requirements likely to be changed.
  + Determination of requirements at an early stage
    - The requirements are relatively well defined, but this is a new product that no one have done it before so I think there are still some requirements that need further clarification.

1. **Development team**
   * Team size:
     + The context referred: “this project consisting of 6 developers and 2 QAs.” Hence, we can conclude that this project is average size, not too big and the team size is also average size.
   * Level of understanding of user requirements by the developers:
     + The development team know a little bit about the situation that need be solve because this is the first time that human facing the large negative effect pandemic like this. So I think the development team hasn’t tried or done some software like this.
2. **User involvement**

* Since the project itself is user-centered, I think users play an important role. In order to bring the best experience to the customers, the shop should include them in the project to evaluate features, UI/UX before giving the project to the General Availability (GA) stage.

🡪 There are several reasons that why I choose **Agile methodologies** especially is **CRUM** framework used in software development:

* **Iterative Development**: The project requires frequent updates and improvements based on user feedback. Scrum's iterative development approach ensures that features are delivered in short cycles (sprints), allowing for continuous enhancements.
* **User-Centric Focus**: Agile emphasizes close collaboration with stakeholders, ensuring that the application meets the needs of parents and babysitters. Regular feedback loops are crucial in achieving this goal.
* **Adaptability**: Start-ups often face changing requirements and priorities. Agile's flexibility allows the team to adjust and reprioritize tasks at the end of each sprint based on emerging needs.
* **Early Delivery**: Agile encourages delivering functional features in each sprint. In this project, this would mean releasing core functionalities early and progressively adding new features.
* **Transparency and Communication**: Scrum ceremonies like Daily Standups and Sprint Reviews promote clear communication within the team and with stakeholders, addressing any issues promptly.

=> Based on the above characteristics, the most suitable software development methodology for this situation is **AGILE** methodology, especially is **CRUM** framework. The advantage of this method is that it’s the combination of the incremental and iterative approach, which mean the project would be built by each feature in a small portion, then gradually adding features and increasing their completeness. It also can assist customers in deploying the product early and collecting reviews and feedback from users to improve the product better. Moreover, this is an average-scale project with not so big team size so more flexibility and dynamic would satisfies with Agile method.

**Question 2:**

Because the team has 6 developers and 2 QAs so I suggest using both black-box testing and white-box testing.

* **White-Box** Testing by Developers: Developers can utilize white-box testing techniques, such as unit testing and integration testing, to validate individual components and their interactions. This helps catch coding errors early in the development lifecycle.
* **Black-Box** Testing by QA: QA professionals can leverage black-box testing techniques to ensure that the virtual fitting room, user reviews, styling suggestions, social sharing, and notifications work as intended. They verify that the software meets business requirements and delivers the desired user experience.

**Benefits of Collaboration:**

When developers and QA professionals collaborate, they contribute to a comprehensive testing strategy that encompasses both internal logic (white-box) and user-centric behavior (black-box). This collaborative effort helps identify a wider range of defects, ensures code quality, and delivers a reliable software product that aligns with user expectations.

**In summary**, white-box testing is more suitable for developers due to their understanding of the code's internal logic, while black-box testing is a primary focus for QA professionals as they assess the software's functionality and user experience. Collaborating on both fronts contributes to a robust and high-quality software product.

**Question 3:**

**Four test cases for what I expected the testing team to use:**

* Test the real-time camera feed feature for live fitting.
* Validate that changes to clothing items reflect instantly in the AR/VR environment.
* Test the accuracy of recommendations when combining multiple items.
* Test that users can submit their reviews and ratings for products they have purchased.

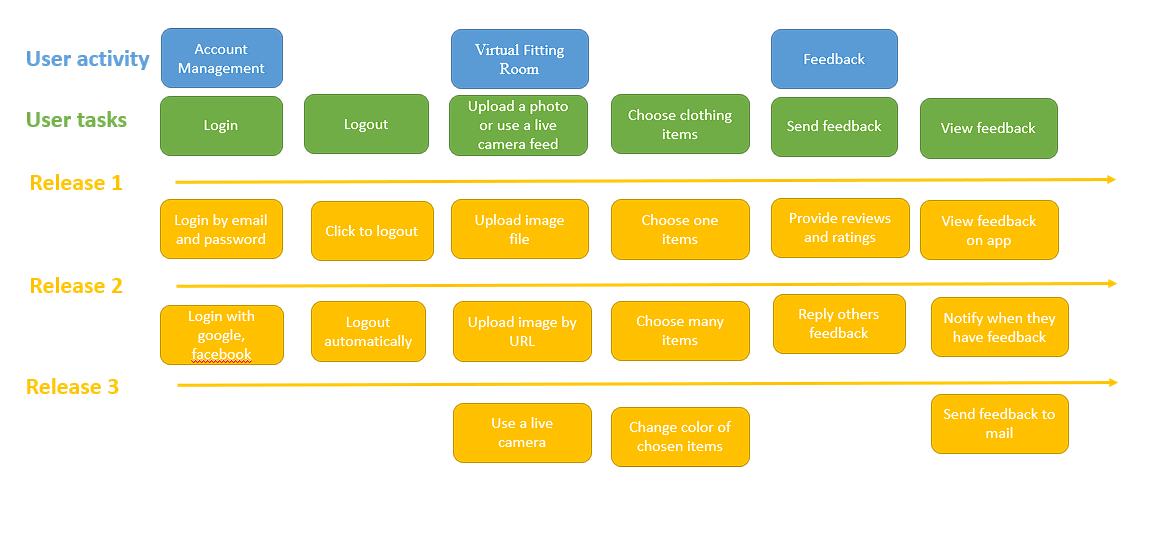
**Question 4:**

* Four functional requirements:
  + Allow users to try on clothes virtually try on clothes using AR/VR technology.
* The app should provide styling suggestions and outfit inspirations to users.
* Users should be able to submit reviews and ratings for products they have purchased.
* Users should receive notifications and alerts for discounts.
* Two non-functional requirements that I recommend:
  + The module should be able to load and display large number of tasks without significant delay.
* The app's user interface should be responsive and optimized for various devices and screen sizes.

**Question 5: Two user stories for this system are:**

* As a user, I want to submit reviews and ratings for products they have purchased so that I can give my feedback for the shop.
* As a user, I want to receive notifications and alerts for discounts so that I can buy clothes with the best price.

**Question 6:**



**Question 7:**

**Assumption 1:** Users will find the Virtual Fitting Room experience engaging and accurate.

* Risk Assessment: High impact if wrong, High probability of being wrong.
* Explanation: If the user buys clothes but it isn’t like in virtual fitting room, users might lose trust in the feature, impacting user satisfaction and purchase decisions. The probability of being wrong is high because virtual fitting room is a new technology so hard to avoid mistakes.

**Assumption 2:** The AR/VR technology will seamlessly integrate with various mobile devices

* Risk Assessment: High impact if wrong, Low probability of being wrong
* Explanation: The risk is that integration issues could arise, affecting a significant portion of users and leading to frustration and potentially discouraging usage.

**Assumption 3:** The assumption is that users will prefer the virtual fitting room feature over traditional browsing.

* Risk Assessment: Low impact of deception, High probability of deception
* Explanation: The risk here is lower, as user preferences might shift gradually, impacting engagement rather than the core functionality of the app. Because the convenience of this app if it can used well, the probability of being wrong is high