**Question 1:**

The software development method I would recommend for this situation is Agile. Because this is a quite complex project, the requirements are not clear and need to discuss further with the customer during implementation. Therefore, the Agile method is the right method for this situation.

Here are some of the characteristics of this situation that make agile a good choice:

(Phương pháp phát triển phần mềm mà tôi muốn đề xuất cho tình huống này là Agile. Vì đây là một dự án khá phức tạp, các yêu cầu chưa được rõ ràng và cần trao đổi thêm với khách hàng trong quá trình thực hiện. Do đó, phương pháp Agile là phương pháp phù hợp cho tình huống này.

Dưới đây là một số đặc điểm của tình huống này khiến cho Agile trở thành một lựa chọn tốt:)

1. **Requirements characteristics**
   * Reliability:
     + It was stated clearly above that there is a problem that needs to be overcome, and an application is required to overcome this problem.
     + • Project requirements are being clearly defined step by step and feasible.
     + This project is possible to run and deploy when completed.

=> The project has a high reliability.

(• Độ tin cậy:

• Ở trên đã nêu rõ rằng có một vấn đề cần phải khắc phục và cần phải có đơn đăng ký để khắc phục vấn đề này.

• Các yêu cầu của dự án đang được xác định rõ ràng từng bước và khả thi.

• Dự án này có thể chạy và triển khai khi hoàn thành.

=> Dự án có độ tin cậy cao.)

* Types and number of requirement:
  + The software requirements contain both *functional requirements* and *non-functional requirements*.
  + All requirements are clearly and easily defined.
  + There are more than 7 requirements that were listed above. It includes 5 functional requirements and 2 non-functional requirements.

=> Types and number of requirements are defined and this project is not too complex for the team.

(• Loại và số lượng yêu cầu:

• Yêu cầu phần mềm bao gồm cả yêu cầu chức năng và yêu cầu phi chức năng.

• Mọi yêu cầu đều được xác định rõ ràng và dễ dàng.

• Có hơn 7 yêu cầu được liệt kê ở trên. Nó bao gồm 5 yêu cầu chức năng và 2 yêu cầu phi chức năng.

=> Loại và số lượng yêu cầu được xác định và dự án này không quá phức tạp đối với nhóm.)

* How often the requirement can change:
  + The requirements that were mentioned above are just some features of mobile application.
  + In the process of developing the project, (Covid 19 situation will change quickly) or(with technology growing rapidly,regularly), some features can be modified to meet the requirements of the customer.

=> The requirements may be changed regularly in the future.

(• Tần suất yêu cầu có thể thay đổi:

• Các yêu cầu nêu trên chỉ là một số tính năng của ứng dụng di động.

• Trong quá trình phát triển dự án, (tình hình dịch Covid 19 sẽ thay đổi nhanh chóng) or (với việc công nghệ ngày càng phát triển mạnh mẽ), một số tính năng có thể được sửa đổi để đáp ứng yêu cầu của khách hàng.

=> Các yêu cầu có thể được thay đổi thường xuyên trong tương lai.)

* Determination of requirements at an early stage
  + Some of the requirements are expressed in great detail above but it isn’t enough to build a completed system.
  + The organization can add or remove some features in the process of the project.

=> It is well-defined but not enough.

(• Xác định các yêu cầu ở giai đoạn đầu

• Một số yêu cầu được trình bày rất chi tiết ở trên nhưng vẫn chưa đủ để xây dựng một hệ thống hoàn chỉnh.

• Tổ chức có thể thêm hoặc bớt một số tính năng trong quá trình thực hiện dự án.

=> Đã xác định rõ nhưng chưa đủ.)

* **Development team**
* Team size:
  + The situation above mentioned our team has 9 people.
  + It is *6 developers, 2 QA* and a product owner who is me.
  + (• Nhóm phát triển
  + • Kích thước nhóm:
  + • Tình huống nêu trên đội chúng tôi có 9 người.
  + • Đó là 6 nhà phát triển, 2 QA và một chủ sở hữu sản phẩm là tôi.)

1 người - < 1 công viec : team lon

1 người - 1 cv : team trung binh

1 nguoi - >1.5 cv: TEAM NHO

=> It is an average team size and enough to build a project that was not too complex.

(=> Đó là quy mô nhóm trung bình và đủ để xây dựng một dự án không quá phức tạp.)

)

* Level of understanding of user requirements by the developers:
  + All of the requirements defined clearly above and our members can understand.
  + The organization can provide additional resources and information when needed.

=> Our team can easily understand and build an application that meets the requirements.

(• Mức độ hiểu biết về yêu cầu của người dùng của nhà phát triển:

• Tất cả các yêu cầu được xác định rõ ràng ở trên và các thành viên của chúng tôi đều có thể hiểu được.

• Tổ chức có thể cung cấp thêm nguồn lực và thông tin khi cần thiết.

=> Nhóm của chúng tôi có thể dễ dàng hiểu và xây dựng được ứng dụng đáp ứng được yêu cầu.)

* **User involvement**
* The situation mentioned “*The organization had contracted with a local company to provide additional resources when needed*.”
* We have a contact with the organization to communicate and give feedback about the project.
* Users can exchange their wishes with us to build the project together

=> The user involvement is high.

(• Sự tham gia của người sử dụng

• Tình huống đề cập “Tổ chức đã ký hợp đồng với một công ty địa phương để cung cấp thêm nguồn lực khi cần thiết.”

• Chúng tôi có liên hệ với tổ chức để trao đổi và đưa ra phản hồi về dự án.

* Người dùng có thể trao đổi, mong muốn của họ với chúng tôi để cùng nhau xây dựng dự án

=> Sự tham gia của người dùng cao.)

To conclude, based on the characteristics requirements that I listed above, I suggest use **Agile/Scrum methodology.** The system must be delivered earlier for the user to buy quickly in Covid-19 context, therefore they should deliver part of the project earlier to get feedback from the user, then apply necessary change for another part. This customer-centric approach ensures that the final product meets the requirements of its users. Overall, the Agile/Scrum methodology is well-suited for this software development project and will likely result in a high-quality end product.

(Để kết luận, dựa trên các yêu cầu về đặc điểm mà tôi đã liệt kê ở trên, tôi khuyên bạn nên sử dụng phương pháp Agile/Scrum. Hệ thống phải được bàn giao sớm hơn để người dùng mua nhanh trong bối cảnh Covid-19, do đó họ nên bàn giao một phần dự án sớm hơn để nhận phản hồi từ người dùng, sau đó áp dụng những thay đổi cần thiết cho phần khác. Cách tiếp cận lấy khách hàng làm trung tâm này đảm bảo rằng sản phẩm cuối cùng đáp ứng yêu cầu của người dùng. Nhìn chung, phương pháp Agile/Scrum rất phù hợp cho dự án phát triển phần mềm này và có khả năng mang lại sản phẩm cuối cùng chất lượng cao.)

**Question 2:**

There is *no mention of the tester's knowledge or experience* in the project description, making black-box testing a suitable option. Furthermore, this type of testing does not require specialized expertise from the analyst, as detailed technical knowledge of the system is not necessary. Therefore, I suggest that the team use **black-box testing** as it allows for user involvement in identifying and providing feedback, specifically with regards to the system's usability.

* *Chọn black-box:* 
  + I suggest that the team use black-box testing because users would be involved in evaluating the project's flaws and providing feedback, particularly regarding the system's usability. In addition, the tester's information and experience are absent from the project description. Additionally, black-box testing permit analyzer can be non-specialized and There is no requirement for the analyzer to have itemized practical information on framework.
  + (+ Tôi đề nghị nhóm sử dụng thử nghiệm hộp đen vì người dùng sẽ tham gia vào việc đánh giá các sai sót của dự án và đưa ra phản hồi, đặc biệt là về khả năng sử dụng của hệ thống. Ngoài ra, thông tin và kinh nghiệm của người thử nghiệm không có trong phần mô tả dự án. Ngoài ra, máy phân tích giấy phép thử nghiệm hộp đen có thể không chuyên dụng và không yêu cầu máy phân tích phải có thông tin thực tế được chia thành từng khoản trên khuôn khổ.)
* *Chọn white-box:*
  + I recommend using white-box testing for this project because team have both developers and testers on your team. This approach allows for collaboration and knowledge sharing between the two groups, enabling developers to identify potential issues and areas for optimization, while testers can provide valuable feedback on user experience and interface. Together, they can ensure thorough testing from both a functional and technical standpoint, resulting in a more robust and reliable product while reducing overall cost and time required for testing.

(+ Tôi khuyên bạn nên sử dụng thử nghiệm hộp trắng cho dự án này vì nhóm của bạn có cả nhà phát triển và người thử nghiệm. Cách tiếp cận này cho phép cộng tác và chia sẻ kiến ​​thức giữa hai nhóm, cho phép các nhà phát triển xác định các vấn đề và lĩnh vực tiềm ẩn để tối ưu hóa, trong khi người thử nghiệm có thể cung cấp phản hồi có giá trị về trải nghiệm và giao diện người dùng. Cùng nhau, họ có thể đảm bảo việc kiểm tra kỹ lưỡng từ cả quan điểm chức năng và kỹ thuật, tạo ra một sản phẩm mạnh mẽ và đáng tin cậy hơn đồng thời giảm tổng chi phí và thời gian cần thiết cho việc kiểm tra.)

**Question 4:**

* The four functional requirements of system are:
  + User can upload a photo or use a live camera feed to try on clothes.
  + User can implement a review and provide feedback on the product and share with others .
  + User can visualize products that are suggested based on individual items of user.
  + Managers can push notifications or email alerts to users about order confirmations, shipment updates, discounts, or new product arrivals.
* The two non-functional requirements of system are:
  + The virtual fitting rooms using augmented reality **(AR)** or virtual reality **(VR)** technologies.
  + The overall design of the system has to take into account both safety and privacy concerns.

**Question 5:**

The two user stories for this system are:

* As a customers, I want to share product to my friends so that I can discuss the product with my friend

As a manager, I want to send emails to all customers so that I can notify all new products to customers, shipment updates and discounts to help customers give more options for buy my products..

**Question 7:**

The three assumptions regarding the Virtual Fitting Room feature are:

* **Upload photo** feature is high impact if wrong, low probability of it being wrong.
  + It is needed for the user when the camera gets at risk.
  + It helps users select old items to compare with new products.
* **Camera feed** feature is high impact if wrong, high probability of it being wrong.
  + Besides uploading photos, there is also a high impact to this requirement because if we do not use camera feed then we do not know the reality of the product when we try it on the mobile phone.
  + Some phones can be dangerous on camera so in the future there is a high probability of it being wrong.
* **Save photo** feature is low impact if wrong, low probability of it being wrong.

**+** Save photo feature can  compare old try of user to get correct decision when buying

**+** This is optional feature, user can not be use

AR OR VR TECHNOLOGY FEATURE IS  is high impact if wrong, low probability of it being wrong.

CAUSE:

+It is needed for the requirement

+

Low Impact, Low Probability

Situation: Inaccurate Billing Information

Cause: Rare occurrences of errors in billing data entry, often stemming from user mistakes or technical glitches. The likelihood of this happening is minimal. The consequences are minor, as any necessary corrections can be implemented through customer service, and the financial impact is negligible.

High Impact, Low Probability

Situation: Data Corruption in Reports

Cause: Infrequent instances of data corruption or formatting problems in generated reports, with a low chance of occurrence. The substantial impact results from the potential misinterpretation of crucial information by users, leading to erroneous decisions and actions.

High Impact, High Probability

Situation: Login Server Downtime

Cause: Frequent short-term outages of the login server, making the likelihood of such events high. The significant impact stems from user dissatisfaction and disrupted workflows due to the inability to access the system during these downtime periods.

(Tác động thấp, xác suất thấp

Tình huống: Thông tin thanh toán không chính xác

Nguyên nhân: Hiếm khi xảy ra lỗi nhập dữ liệu hóa đơn, thường xuất phát từ sai sót của người dùng hoặc trục trặc kỹ thuật. Khả năng điều này xảy ra là tối thiểu. Hậu quả là nhỏ, vì bất kỳ sự điều chỉnh cần thiết nào cũng có thể được thực hiện thông qua dịch vụ khách hàng và tác động tài chính là không đáng kể.

Tác động cao, xác suất thấp

Tình huống: Sai dữ liệu trong báo cáo

Nguyên nhân: Các trường hợp hỏng dữ liệu hoặc sự cố định dạng không thường xuyên trong các báo cáo được tạo, với khả năng xảy ra thấp. Tác động đáng kể là kết quả của việc người dùng có thể hiểu sai thông tin quan trọng, dẫn đến các quyết định và hành động sai lầm.

Tác động cao, xác suất cao

Tình huống: Thời gian ngừng hoạt động của máy chủ đăng nhập

Nguyên nhân: Máy chủ đăng nhập thường xuyên ngừng hoạt động trong thời gian ngắn khiến khả năng xảy ra các sự kiện như vậy cao. Tác động đáng kể bắt nguồn từ sự không hài lòng của người dùng và quy trình công việc bị gián đoạn do không thể truy cập hệ thống trong những khoảng thời gian ngừng hoạt động này.)

Low Impact, Low Probability

Situation: Intermittent User Interface Glitches

Cause: Infrequent incidents of minor glitches within the user interface, such as temporary visual discrepancies or graphical anomalies, with a low likelihood of occurrence. The effect is minor as these glitches are short-lived and do not impede users' capacity to complete tasks. Users can smoothly continue utilizing the software without substantial interruptions.

(Tác động thấp, xác suất thấp

Tình huống: Trục trặc giao diện người dùng không liên tục

Nguyên nhân: Các sự cố không thường xuyên xảy ra với các trục trặc nhỏ trong giao diện người dùng, chẳng hạn như sự khác biệt tạm thời về hình ảnh hoặc các điểm bất thường về đồ họa, với khả năng xảy ra thấp. Ảnh hưởng không đáng kể vì những trục trặc này chỉ tồn tại trong thời gian ngắn và không cản trở khả năng hoàn thành nhiệm vụ của người dùng. Người dùng có thể tiếp tục sử dụng phần mềm một cách suôn sẻ mà không bị gián đoạn đáng kể.)

High Impact, Low Probability

Situation: Unexpected Breakdown in Integration

Cause: Infrequent cases of integration breakdowns between software components, usually triggered by factors such as version disparities, with a slim chance of occurrence. The notable consequence results from interrupted data exchange and inter-system communication, potentially leading to delays and inaccuracies.

(Tác động cao, xác suất thấp

Tình huống: Sự cố bất ngờ trong quá trình tích hợp

Nguyên nhân: Các trường hợp không thường xuyên xảy ra sự cố tích hợp giữa các thành phần phần mềm, thường được gây ra bởi các yếu tố như sự khác biệt về phiên bản, với khả năng xảy ra rất nhỏ. Hậu quả đáng chú ý là việc trao đổi dữ liệu và liên lạc giữa các hệ thống bị gián đoạn, có khả năng dẫn đến sự chậm trễ và thiếu chính xác.)

High Impact, High Probability

Situation: Sluggish System Performance During Busiest Periods

Cause: Regular occurrences of system slowdowns and delayed responses encountered by users during peak usage hours, resulting in a strong likelihood of this situation happening. The notable impact arises from reduced efficiency and user discontent caused by extended waiting periods.

(Tác động cao, xác suất cao

Tình huống: Hiệu suất hệ thống chậm chạp trong thời gian bận rộn nhất

Nguyên nhân: Hệ thống thường xuyên bị chậm và phản hồi chậm mà người dùng gặp phải trong giờ sử dụng cao điểm, dẫn đến rất có thể tình huống này sẽ xảy ra. Tác động đáng chú ý phát sinh từ việc giảm hiệu quả và sự bất mãn của người dùng do thời gian chờ đợi kéo dài.)

Performance:

* **Response Time:** The system should respond to user interactions within 2 seconds.
* **Throughput:** The system should support a minimum of 1000 concurrent users.
* **Latency:** The system's communication latency between components should not exceed 50 milliseconds.

2.        Reliability:

* **Availability:** The system should be available 99.9% of the time.
* **Fault Tolerance:** The system should continue functioning properly even in the presence of hardware or software failures.
* **Recoverability:** The system should be able to recover data and resume normal operation within 1 hour after a failure.

3.        Security:

* **Authentication:** Users must be authenticated using two-factor authentication before accessing sensitive data.
* **Data Encryption:** All sensitive data should be encrypted both in transit and at rest.
* **Access Control:** Different user roles should have appropriate access privileges to system features and data.

4.        Usability:

* **User Interface Consistency:** The user interface should follow the company's branding guidelines and maintain a consistent design.
* **Accessibility:** The system should comply with accessibility standards (e.g., WCAG) to ensure usability for users with disabilities.
* **User Training:** The system should be intuitive and require minimal user training to operate.

5.        Scalability:

* **Vertical Scalability:** The system should be able to handle a 50% increase in load by adding more resources (e.g., CPU, memory).
* **Horizontal Scalability:** The system should be compatible with the Windows, macOS, and Linux operating systems.

6.        Maintainability:

* **Modularity:** The system's components should be organised into separate modules to facilitate easy maintenance and updates.
* **Code Documentation:** All code should be well-documented with comments and clear explanations of functionality.
* **Extensibility:** The system's architecture should allow for easy integration of new features and functionalities.

7.        Compatibility:

* **Browser Compatibility:** The system should be compatible with the latest versions of popular web browsers (e.g., Chrome, Firefox, Safari).
* **Platform Compatibility:** The system should run on Windows, macOS, and Linux operating systems.

8.        Performance Efficiency:

* **Resource Utilisation:** The system should use no more than 50% of available system resources under typical load conditions.
* **Energy Efficiency:** The system should minimise energy consumption to reduce environmental impact.

(Hiệu suất:

Thời gian phản hồi: Hệ thống sẽ phản hồi các tương tác của người dùng trong vòng 2 giây.

Thông lượng: Hệ thống phải hỗ trợ tối thiểu 1000 người dùng đồng thời.

Độ trễ: Độ trễ giao tiếp của hệ thống giữa các thành phần không được vượt quá 50 mili giây.

2. Độ tin cậy:

Tính khả dụng: Hệ thống phải khả dụng 99,9% thời gian.

Khả năng chịu lỗi: Hệ thống sẽ tiếp tục hoạt động bình thường ngay cả khi có lỗi phần cứng hoặc phần mềm.

Khả năng khôi phục: Hệ thống có thể khôi phục dữ liệu và hoạt động trở lại bình thường trong vòng 1 giờ sau khi xảy ra sự cố.

3. Bảo mật:

Xác thực: Người dùng phải được xác thực bằng xác thực hai yếu tố trước khi truy cập dữ liệu nhạy cảm.

Mã hóa dữ liệu: Tất cả dữ liệu nhạy cảm phải được mã hóa cả khi truyền và khi lưu trữ.

Kiểm soát truy cập: Các vai trò người dùng khác nhau phải có đặc quyền truy cập thích hợp vào các tính năng và dữ liệu của hệ thống.

4. Khả năng sử dụng:

Tính nhất quán của giao diện người dùng: Giao diện người dùng phải tuân theo các nguyên tắc xây dựng thương hiệu của công ty và duy trì một thiết kế nhất quán.

Khả năng tiếp cận: Hệ thống phải tuân thủ các tiêu chuẩn về khả năng tiếp cận (ví dụ: WCAG) để đảm bảo khả năng sử dụng cho người dùng khuyết tật.

Đào tạo người dùng: Hệ thống phải trực quan và yêu cầu đào tạo người dùng tối thiểu để vận hành.

5. Khả năng mở rộng:

Khả năng mở rộng theo chiều dọc: Hệ thống sẽ có thể xử lý tải tăng 50% bằng cách thêm nhiều tài nguyên hơn (ví dụ: CPU, bộ nhớ).

Khả năng mở rộng theo chiều ngang: Hệ thống nên hỗ trợ thêm các nút bổ sung vào cụm để xử lý tải tăng lên.

6. Khả năng bảo trì:

Tính mô đun: Các thành phần của hệ thống nên được tổ chức thành các mô đun riêng biệt để tạo điều kiện bảo trì và cập nhật dễ dàng.

Tài liệu mã: Tất cả mã phải được ghi lại đầy đủ với các nhận xét và giải thích rõ ràng về chức năng.

Khả năng mở rộng: Kiến trúc của hệ thống sẽ cho phép tích hợp dễ dàng các tính năng và chức năng mới.

7. Khả năng tương thích:

Khả năng tương thích với trình duyệt: Hệ thống phải tương thích với các phiên bản mới nhất của các trình duyệt web phổ biến (ví dụ: Chrome, Firefox, Safari).

Khả năng tương thích nền tảng: Hệ thống sẽ chạy trên các hệ điều hành Windows, macOS và Linux.

8. Hiệu quả hoạt động:

Sử dụng tài nguyên: Hệ thống nên sử dụng không quá 50% tài nguyên hệ thống có sẵn trong các điều kiện tải điển hình.

Hiệu quả năng lượng: Hệ thống nên giảm thiểu mức tiêu thụ năng lượng để giảm tác động đến môi trường.)