**ANH EM NHỚ CHỈNH LẠI CHO PHÙ HỢP HOÀN CẢNH VÀ YÊU CẦU CỦA ĐỀ, TẤT CẢ TRƯỜNG HỢP CỦA ĐỀ PHẢI CÓ DÍNH DÁNG TRONG LUẬN ĐIỂM**

**ĐỀ COVID19**

**Question 1:**

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.
     + The project requirements are well-defined and possible.
     + This project is possible to run when completed.

=> The project has a high reliability.

* + 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.

* + How often the requirement can change:
    - The requirements that were mentioned above are just some features of this application.
    - In the process of the project, Covid 19 situation will change fast, some features can be modified and changed to meet the requirements of the customer.

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

* + 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.

* Development team
  + Team size:
    - The situation above mentioned our team has 9 people.
    - It is *6 developers, 2 QA* and a leader team who is me.

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.

* + 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.

* 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.

=> The user involvement is high.

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.

**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.

**Question 3:**

* The four test case I would to suggest to test team :

+ **User-experience testing**: Since this app is particularly for manager so we need to clear what he wants ,it is always good to see satisfaction from the manager with the product

+ **Security testing**: Because all of the information feedback are send to the manager so the dev team has to make sure there are NO issues in securing users information leaked out to the customer.

+ **Black-box testing**: It allows for user involvement in identifying and providing feedback, specifically with regards to the system's usability

+ **White-box testing**: Because the team has both developers and testers on your team. 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.

-> 4 test case use for this situation :

Test case 1 ( **Usability Test Case ) ( tính tương thích ( mobile )**

input : sử dụng app ở máy tính để bàn

output : chỉ sử dụng được mỗi tính năng upload photo và không sử dụng được camera feed đối với

Test case 2 ( **Performance Test Case ) ( hiệu suất )**

**input: có 10000 người sử dụng app**

**output: app sẽ không hoạt động**

**Test case 3: User Interface Test Case**

**input: cái nút bấm chọn ảnh quá nhỏ**

**output: người dùng sẽ chọn nhầm hoặc không bấm được**

**input: trường hợp**

**output:kết qủa tệ của trường hợp đó**

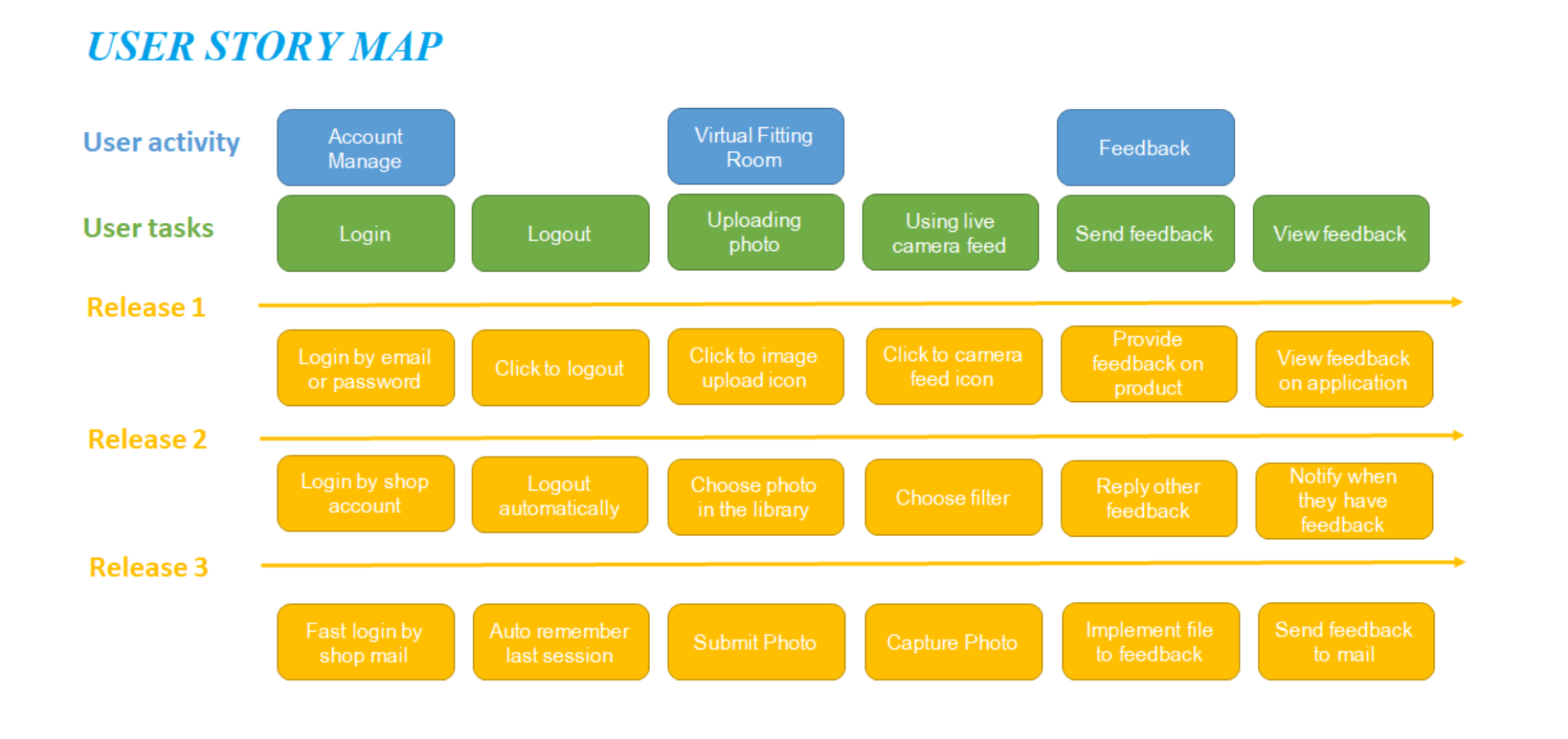
**Question 4:**

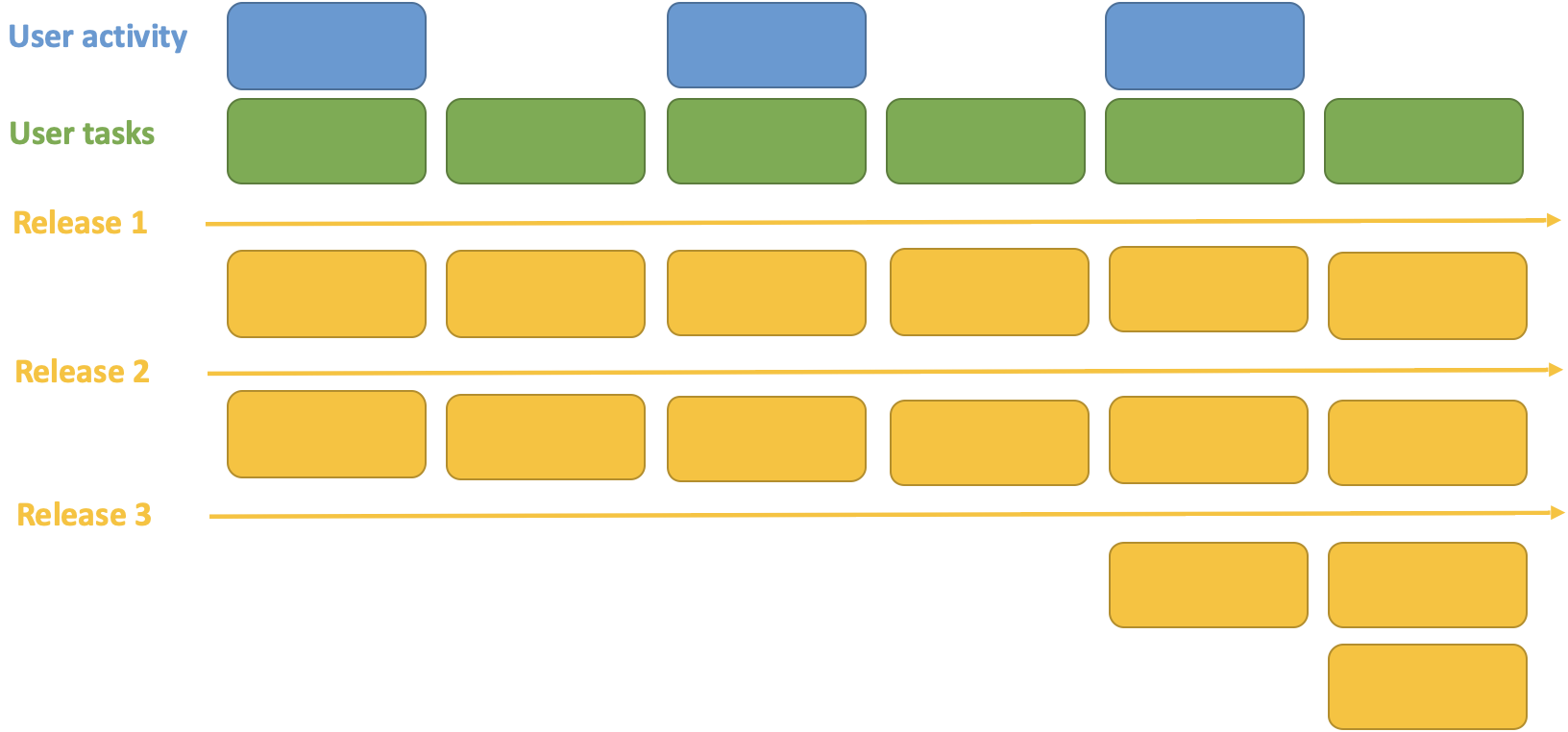
* The four functional requirements of system are:
  + User can upload a photo or use a live camera feed to try on clothes.
  + User can implement a review and provide feedback on the product and share with others .
  + User can visualize products that are suggested based on individual items of user.
  + 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 user, I want to send feedback to shop and share my friends about product so that shop can adjust or improve product in the future and I want my friends can take a correct decision before buy product.
  + As a manager, I want to send emails to all customers so that I can notify all new products, shipment updates and discounts to help customers give more selection.

**Question 6:**





**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.

Giờ cao điểm

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.

Lỗi version

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.

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.