Requirements Management

Greenwich University of Vietnam

Coursework Term 2

Learner name: Nguyễn Minh Quân.

ID learner: GCS18644.

Class: TCS2102A.

Assessor name: Le Ngoc Thanh.

Assignment due: Assignment submitted: 27/04/2022

# **Section A - Produce an Executive Summary**

1. Introduction.

Movie Ville (MV) is a traditional cinema that has not yet applied modern technologies and is on the verge of falling behind since the video streaming platform was born. To combat the collapse of the whole business, the business successor David has improved the model of this business by applying modern technologies and operating with the previous traditional system. Specifically, David came up with an idea for operating a paperless ticket sales system through a mobile application platform and a website. Gunner Satumo a consultant was hired by David, suggesting that the mobile application for digital movie tickets would be the first project to execute. After many meetings were held, Gunner adopted the Agile model for development, and in particular he felt that DSDM would be an appropriate framework for the development of this system. The method proposed by Gunner is really suitable for the current situation of the business because of its quickness. Businesses need a website that can be quickly initialized for online ticket sales that can be run on smartphones. The basic functions required are: allowing people to select a movie and buy tickets for that movie by specifying the number of seats, creating a digital ticket for all purchased tickets, allowing anyone people select a movie, then show a short summary, allowing people to rate a movie after watching. The app allows people to pay for their transactions in standard ways, such as credit cards and PayPal. This system will be implemented according to Agile method which will get the system out as soon as possible including the basic functions outlined above. Given the agile nature of the Agile methodology, the proposed system still has some limitations, but there is room for improvement in the future.

# **Section B – High level requirements analysis and MoSCoW prioritization**

In this section I will analyze the high-level requirements and not high-level requirements in the given scenario. I will first briefly outline the high-level requirements.

In project management, requirements are the summary of the tasks and conditions that need to be met for the project to be successful. Requirements come in a wide variety of categories including business requirements and technical requirements. While the business requirement outlines what the organization is expected to do as a result of the project, the technical requirement dictates specific features or customer functionality.

High-level requirements emphasize the "what" of the project rather than the "how". The senior request does not provide details on each stage of the membership or missions. A Simple High Level Requirement is a document that is easily accessible and simple for stakeholders to understand. High level requirements should be a functional requirements.

The assumption I am making in this scenario is that an online ticketing system product capable of operating on a website and on a smartphone platform is announced **within 3 months** of the project's inception.

**Functional requirements:** are features or functions that developers must build in order for users to complete their duties. As a result, it's critical to make them apparent to both the development team and the stakeholders. In general, functional requirements explain how a system behaves under certain situations.

**Nonfunctional requirements:** do not involve system functionality, but rather outline how the system will operate.

## B1 **User stories that are not appropriate high level requirements.**

Based on what Mr. Gunner Satumo outlined in the script, I will list the functions according to each type of High-level requirement or not High-level requirement.

|  |  |  |
| --- | --- | --- |
| **User stories ID** | **User stories** | **Reason** |
| 1 | As a product owner, I want the interface must be sleek. It’s not flashy but has to be elegant to show that we have flair. | This is not a high-level requirement because it doesn't really matter to the problem the business is facing. A moderate interface that is built quickly to respond to the current situation of the business will be more suitable. Besides, the Interface can be improved later and repeated many times. |
| 2 | As a product owner, I want to know exactly where my patrons come from. | This is also not a high-level requirement because the only focus is on improving ticket sales and adding an online ticket sales model. Besides, just starting an online ticketing system, we can't really predict that many people will become patrons or not. |
| 3 | As an Interior Designer, I want the interface has the look and feel of the app similar to the color schemes of the theatre’s interior to preserve the branding. | This is not a high-level requirement either. As mentioned in user story id 1, what we need is a system that quickly publishes and operates digital ticketing. |
| 4 | As a customer, I want to receive a notification when the latest film is published. | This is also not a high-level requirement, the latest movies will be automatically announced in the main interface as well as posters posted outside the cinema. In addition, trailers of upcoming movies will be advertised in the opening minutes before the movie is shown. So creating a function that notifies the user for a new movie will not really be necessary at the beginning. |
| 5 | As a customer, I want to give the feedback about the facility improvements. | This is also not a high-level requirement because feedback is always the final destination of a system. As the scenario suggests, we're at the beginning of a system where the focus is on digital ticket sales. This can be improved after the basic functions are complete. |
| 6 | As a Ticket Sales, I think a counter ticket sales should be conducted. | This is also not a high-level requirement because time is a barrier to this. We cannot run 2 applications at the same time in a short period of time. Besides, the complexity of synchronizing databases across two different applications is also a big challenge. |
| 7 | As a customer, I want to print my PDF download of the ticket. | This is also not a high-level requirement because it is possible to replace the printed PDF with the use of digital tickets. |
| 8 | As a Ticket Sales, We don't need to create a website but just build an application because the number of users of the application is more than the website. | This is also not a high-level requirement because the initial goal of the scenario is a system that runs on both web and application platforms. Therefore, this idea is contrary to the stated goal and cannot be implemented. |
| 9 | As an Accountant, I want to have the monthly reports on this system to see the number of bookings for each showing so I can draw statistics from this. | Without a report function, Accountant just takes some time for doing the calculation manually. It’s not affect to the ticket sale revenue. Therefore, it’s not a high-level requirement to me. |
| 10 | As a Concession, I want to have user purchase their snacks on the app too and serve it to them in the theatre once they are seated. | This is also not a high-level requirement because as the main purpose stated in the scenario, we are focusing on improving ticket sales performance. We can still continue to sell food and drinks over the counter and don't necessarily sell them online. |
| 11 | As a Concession, I want the customers are able to noting their favorite snacks on a system, so they automatically order it when they buy the ticket. | Without this function, the digital ticket function still work. Besides, this function can make a mistake to customer when they do not mean to purchase snacks .Therefore, it’s not a high-level requirement to me. |
| 12 | As a Concession, I want this app allow customers to make notes on the movies they’ve seen, so they can look at these again. | This is also not a high-level requirement because to me it is redundant and unrelated to improving ticket sales. |

## **B2. User stories that are appropriate high level requirements.**

|  |  |  |
| --- | --- | --- |
| **User stories ID** | **User stories** | **Reason** |
| 1 | As a customer, I need an account to purchase for the ticket I want. | This is a functional requirement and of course a high-level requirement because to be able to pay for purchased movie tickets, an account is required to identify who the user is. At the same time the account created is considered a member and can accumulate points for the account. Therefore, it is likely that users will buy movie tickets for the next time and this helps increase ticket sales that businesses are facing. |
| 2 | As a product owner, I want my application is error free. | This is a nonfunctional requirement and of course a high level requirement because there is nothing more frustrating than a buggy application. This reduces performance and user interaction. |
| 3 | As a customer, I want to choose my seats when make the booking. | This is a functional requirement and of course a high-level requirement because it is a must-have function when implementing a booking process. If this function is not completed, it will cause many errors in the processing of the system. This directly affects ticket sales. |
| 4 | As a customer, I want to rate the film after watching and discuss the movie with other customer using app. | This is a functional requirement and of course also a high-level requirement because leaving feedback from users is necessary to help other customers have a good view of the application. New customers tend to go through the reviews first and then experience the system. A function that allows users to rate helps entice more users to increase the likelihood of buying tickets. |
| 5 | As a customer, I need maps of theatre seating areas so that I can choose my favorite seats. | This is an indispensable functional requirement because users need to know exactly where they sit to avoid sitting in the wrong seats of other guests. Therefore, to me this is a high level requirement. |
| 6 | As a customer, I want to choose a movie and buy tickets for that movie by specifying the number of seats. | This is an indispensable functional requirement because without this function users cannot buy tickets online and this completely goes against the original goal. Therefore, to me this is a high level requirement. |
| 7 | As a ticket sales, I think we need to have a backup plan for single seats being left open to eliminate cases where customers buying tickets at the last minute cannot sit close to each other. | This is a functional requirement that directly affects ticket sales because when customers buy tickets at the last minute and they can't sit close together, customers will actively cancel their ticket purchases. The single seats that are left alone will not be booked, leading to a loss of income from these seats. |
| 8 | As a customer, I need a digital ticket for all purchased ticket so that I do not need to wait in line for buying paper ticket. | This is the most important functional requirement because the initial goal set in the scenario is "paperless ticket". Therefore, creating a digital ticket to replace the traditional ticket is indispensable in this system. Therefore, it’s a high level requirement. |
| 9 | As a customer, I want to pay for my transactions by using my credit card or PayPal account. | This is a functional requirement because this indispensable function saves time queuing for tickets and paying for tickets at the counter. Direct payment on smartphones helps users to be more active in paying and getting movie tickets and seats that they like. Therefore, it’s a high level requirement. |
| 10 | As a customer, I want to have rewards after referring friends and buying ticket. | This is a functional requirement because this directly affects the number of users and ticket sales that businesses really need. Accumulating points for rewards later helps businesses retain customers. Besides, the rewards through referring friends are an indirect user enticement. This increases the number of users and loyal customers. |
| 11 | As a product owner, I want this is a security application. | This is a nonfunctional requirement and I consider this requirement very important and necessary because a poor quality system for safety and security is the leading cause of customer or website information leaks or website cannot exist online. This leads to a lot of financial harm and faces legal proceedings that every business does not want to face. |
| 12 | As a product owner, I want my website must load quickly. | This is a nonfunctional requirement because of the following reasons. A website with slow loading speed will create a bad user experience. This will take a lot of time when users interact in some functions such as searching, selecting, and paying. Users are likely not patient enough to wait and they will stop buying tickets and paying. This results in lost customers and reduced ticket sales. So this is a high level requirement. |

## **B3. Moscow/Timeboxing priority.**

The dynamic systems development method (DSDM) is said to be a framework for managing Agile projects that is highly regarded by design professionals in the process of improving the quality of agile application development processes. The benefits of DSDM to the project include that quality, cost and time are strictly managed right from the beginning of the project. Therefore, all the tasks included in the project must be classified according to their importance. This has led to the introduction of specialized prioritization mechanisms.

MoSCoW was used to implement this technique. It is understandable that MoSCow is a simple technique, but rigorous enough to set priorities with or without a timebox. MoSCoW provides better efficiency by giving a certain deadline for each function, task, etc. Besides, to understand what MoSCoW includes, simply MoSCoW stands for: **M**ust have, **S**hould have, **C**ould have, **W**ill not have. However, some businesses expect the "**W**" in MoSCoW to be "**W**ish" or ”**W**antto have”.[[1]](#footnote-1)

The following are the MoSCoW Prioritization Categories.

* **Must-have:** The name has shown the meaning of this category is "Must have" which means indispensable functions and tasks in this section for a system. It is conceivable that this category is the non-negotiable product requirements that include the mandatory to the development team. The product will not run or fail if the requirements in this category are missing.
* **Should-have:** This is a lower level category than "Must-have". It can be understood that the inevitable here is that if the requirements in this section are missing, the product will still run. However, the value that the product brings will not be high. The next thing that distinguishes it from "Must-have" is that initiatives from "Should-have" can be carried out in the future without directly affecting the performance of the current product.
* **Could-have:** In this category, comments are not strictly necessary for the main functionality of the product. It can also be interpreted as "nice-to-have" indicating a very small effect compared to the results that the product gives.
* **Will-not-have:** This category is in place to manage ideas to prevent scope escalation. Ideas in this section will not be prioritized for implementation and there is no specific time frame for functions and tasks.

Based on the requirements in the available scenario and the theories of MoSCoW above, I will provide a table below describing the application of MoSCoW in the given scenario.

### **B3.1. Prioritizing the requirements by using MoSCoW/timeboxing rules.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID Requirements** | **Requirements** | **Priority** | **Business priority (0-9)** | **EST (days)** | **Assign** |
| 3 | As a customer, I want to choose my seats when make the booking. | **Must have** | 9 | 3 | **Manfred Smith** |
| 1 | As a customer, I need an account to purchase for the ticket I want. | 9 | 6 |
| 2 | As a product owner, I want my application is error free. | 9 | 4 |
| 5 | As a customer, I need maps of theatre seating areas so that I can choose my favorite seats. | 9 | 2 |
| 9 | As a customer, I want to pay for my transactions by using my credit card or PayPal account. | 9 | 5 |
| 8 | As a customer, I need a digital ticket for all purchased ticket so that I do not need to wait in line for buying paper ticket. | 9 | 1 |
| 11 | As a product owner, I want this is a security application. | 9 | 10 |
| 7 | As a ticket sales, I think we need to have a backup plan for single seats being left open to eliminate cases where customers buying tickets at the last minute cannot sit close to each other. | **Should have** | 6 | 7 |
| 4 | As a customer, I want to rate the film after watching and discuss the movie with other customer using app. | 5 | 2 |
| 10 | As a customer, I want to have rewards after referring friends and buying ticket. | **Could have** | 4 | 5 |
| 12 | As a product owner, I want my website must load quickly. | **Want to have** | 3 | 1 |
|  | **TOTAL (Days)** | | | **46 days** |

### **B3.2. The reason why I set the priority at B3.1**

The following are the explanations for my analysis when prioritizing the requirements in B3.1. The requirements are prioritized from the highest level ( Must have ) to the lowest level ( Want to have ). In the "W" part of the MoSCoW approach, I put it as "Want to have" because of some requirement that every business wants it but can't put it first.

* **Must have**
* Customers want to choose their seats when make the booking.

This function is the core of the whole system, the main goal to operate the application and website. Without this function => users cannot choose seats => cannot buy tickets => do not improve ticket sales. For all the reasons mentioned, this is a must-have function. And besides, the priority for this function in the business aspect is set to the highest level (9). I anticipate completing this functionality in a 3 day period based on the circumstances of the business.

* Customers must be able to create account and login before booking their seats and purchase.

This is a must-have function in the system to identify who the user is and allow them to pay with what they have booked. Based on the sample reports in [MoSCoW Method - Definition and lots of Helpful Examples*[[2]](#footnote-2).*], the registration and login functions are always in the "Must have" section. Because this is a function that requires the system to have a login function and registration, it must be present before you can use the payment function. Because of that, I predict it will probably take twice as long as the booking function (6 days). And of course this is a must-have function, so the priority is also at the highest level (9).

* As the Managing Director, I want the system has to be errors free.

No one wants to experience a system with lots of unresolved bugs. This is the result of a faulty product that leads to many consequences such as: duplicate bookings by multiple users, inability to pay or unexpected errors, so bugs must be fixed before being released. Besides, the priority is also placed on top so that the offered product is well received by users. In the 3-month product roadmap I anticipate bugs being fixed during implementation. The total time to fix errors can be wrapped up in 4 days.

* As a customer, I need maps of theatre seating areas so that I can choose my favorite seats.

Imagine that you are booking a movie ticket without knowing where you want to sit. Can you choose a position that suits your vision? That's why the system needs a map to determine where you want to sit so that you can book tickets for your preferred location. To me this is a simple and must-have function for the system to work. This function can be completed in 2 days.

* As a customer, I want to pay for my transactions by using my credit card or PayPal account.

The online ticketing system would not be complete without the payment function. The purpose of the business is to apply new technologies to the online ticketing system. Therefore, it is necessary to replace the traditional payment method with direct payment on the system. Once the customer has paid, they cannot return it even though they don't really want to see the movie. This results in significantly improved ticket sales compared to traditional payment methods. Given the complexity of this function, I expect it to be completed within 5 days.

* As a customer, I need a digital ticket for all purchased ticket so that I do not need to wait in line for buying paper ticket.

The goal set out in the given scenario describes that the business is in need of a "paperless ticket" and the digital ticket printing function meets this requirement of the business. Therefore, this is a must-have function in the system. The simplicity of this function should be done in 1 day.

* As a Managing Director, I want this is a security application.

Security is always the top priority of a system. Imagine a scenario with a system with poor security. This proves that the system is vulnerable leading to the system being attacked by external forces. The consequences are loss of customer data or irreparable system failure. As a result, businesses face serious legal problems related to the disclosure of customers' personal information. This directly affects the budget of the business and the business may go bankrupt. So this is a must-have and top-priority function. I will spend a lot of time (10 days) on this critical function that directly affects this business.

* **Should-have**
* As a ticket sales, I think we need to have a backup plan for single seats being left open to eliminate cases where customers buying tickets at the last minute cannot sit close to each other

Without this function, the system is still completely stable with the above mentioned functions. However, without this function the system will not bring high value and the user experience is likely to decrease. Imagine that you are the last guests and cannot sit together because of the lack of this function. These guests will not continue to book tickets and these empty seats will lose ticket sales. So here is a should-have function with a rather high priority (5). However, this is quite a complex function so it takes quite a while to get the function perfect. (7 days)

* As a customer, I want to rate the film after watching and discuss the movie with other customer using app.

This can be considered as a secondary function with the aim of improving ticket sales by allowing customers to have an overview of previous customer reviews. If this function does not exist in the system, the system will not be affected to the ticket purchase and payment process. However, customers' inability to see reviews to see how appropriate a movie is for them leads to customer conflict with the cinema. Therefore, although it does not play a major role in improving ticket sales, the value that this function brings is not small. This is a should-have function for me and in 2 days could get it done.

* **Could-have**
* As a customer, I want to have rewards after referring friends and buying ticket.

This can be considered as a secondary function with the purpose of attracting more users by accumulating points when referring new users. At first glance, it seems that it is not really necessary for the original purpose of the business. However, as mentioned in the scenario, the business owner is on a big budget so this function is an could-have function. It can be done after the previously mentioned higher priority functions have completed. The complexity of this function is high, so within 5 days the development team can apply it to the system.

* **Want-to-have**
* As a Managing Director, I want my website must load quickly.

This can be seen as a high-level requirement with a direct impact on the system. However, the system can still operate stably even though the page load is slow. In a further perspective in the future, this function wants to be applied to the system by business owners when the number of customers booking tickets online increases. So this is a want-to-have function and can be completed in a day.

1. *MoSCoW Prioritization*. (2021, September 11). ProductPlan. <https://www.productplan.com/glossary/moscow-prioritization/> [↑](#footnote-ref-1)
2. *MoSCoW Method - Definition and lots of Helpful Examples.* (n.d.). Consuunt. https://www.consuunt.com/moscow-method/ [↑](#footnote-ref-2)