Team 9

Daniel Wang, Maria Zhang, Trinh To, Phuong Vu, Safwaan Taher

Requirements/Specifications

Part #1: SRS (MoSCoW) requirements

Software Requirements Specification

For

Babs Burger Bonanza (BBB)

Version 1.0 approved

Prepared by Team 9

CSS 370

May 13, 2022

Team 9

Daniel Wang, Maria Zhang, Trinh To, Phuong Vu, Safwaan Taher

Table of Contents

Version History

Name	Date	Reason For Changes	Version

Team 9

Daniel Wang, Maria Zhang, Trinh To, Phuong Vu, Safwaan Taher

1. Introduction

2. Overall Description

3. External Interface Requirements

4. System Features

4.1 Loging in/out for Employee

4.1.1 Description and Priority

4.1.2 Stimulus/Response Sequences

4.1.3 Functional Requirements

- REQ-1A: The application/system interface should be responsive (use Bootstrap).
- REQ-2A: The application/system should display an authentication interface (username, password) every time the user accesses the system.
- REQ-3A: The application/system should be able to connect to a thirdparty two-step authentication application.
- REQ-4A: The system should have a two-step authentication

4.2 Warehouse Operating Features

4.2.1 Description and Priority

4.2.2 Stimulus/Response Sequences

4.2.3 Functional Requirements

- REQ-1B: The system should obtain databases to record all transactions, inventories, and operations

- REQ-2B: The system should send the low stock alert to users of certain roles
- REQ-3B: The system should allow users to input data
- REQ-4:B The system Records Create interface should have a this input field (Date, Related parties, Time,...)
- REQ-5B: The system should have database connect to save all the record of operating activity.
- REQ-6B: The system should categorize data into different record groups (food truck record, ingredient record, schedule record, …)
- REQ-7B: The system should be able to produce formatted reports of sales and ingredients order
- REQ-8B: The system should have a drop down list of all the warehouse category activity.
- REQ-9B: The system should synchronize data from across housewares for emergency delivery.

4.3 Food Truck Operating Features

4.3.1 Description and Priority

4.3.2 Stimulus/Response Sequences

4.3.3 Functional Requirements

- REQ-1C: The system should display the Foodtruck operating schedule by vehicle ID with the chef, driver and cashier on each day.
- REQ-2C: The system should include a POS payment managing system, sale and commission calculation.
- REQ-3C: The system should provide updated and accurate route navigation for truck driver
- REQ-4C: The system must be able to correctly display menu items served by the food truck.

- REQ-5C: The system must be able to be changed to display new menu items as well as remove out of stock items.
- REQ-6C: The system must be able to communicate between warehouse workers and food truck workers.
- REQ-7C: The system must be able to show current and past orders from a certain food truck.
- REQ-8C: The system should allow food truck workers to order new supplies.

4.4 Taking Order System

4.4.1 Description and Priority

4.4.2 Stimulus/Response Sequences

4.4.3 Functional Requirements

- REQ-1D: The system will only allow customers to order within business hours.
- REQ-2D: The system will only allow customers to order the food listed on the app.
- REQ-3D: The system will notify customers in case of order adjustments or cancellations.
- REQ-4D: The system will let the chef know if the customer has special requests for their food.
- REQ-5D: The system will give customers an estimated time when their order is ready.
- REQ-6D: The system will let customers place one order at a time.

4.5 Sales/Business Features

4.5.1 Description and Priority

4.5.2 Stimulus/Response Sequences

4.5.3 Functional Requirements

- REQ-1E: The system should be able to predict location with high benefits based on transaction history
- REQ-2E: The system should be able to handle and secure payment transaction without the help of third party

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- The system will notify customers 30 minutes before closing.
- The system will let customers know their order is ready to be picked up in 5 minutes.

5.2 Safety Requirements

- The payment system will secure the customer's payment method so that the payment will be encrypted and used only on paying for the food.
- The system will have a backup so that all the customers’ orders are saved when the system is down.

5.3 Security Requirements

- The system should secure sensitive, transactional, and personal data so they can only be accessed by intended users
- Different users (e.g. manager, worker, cook,...) have different identities and access.
- The system should eliminate the login or authenticated session if the second authentication step is not done within 3 minutes after the first authentication step

- The length of the password is 6 - 12, contain numbers, letters, and special characters.

5.4 Software Quality Attributes

- The system should be portable such that it can be easily used on multiple platforms such as iOS, Android, Chrome.
- The system must be reliable such that even if the system is flooded with requests/purchases, the system will still function as well as under no load.
- The system must work with high performance such that clicks, outputs, inputs, purchases, requests, and other functions in the system are responsive and fast.
- The system should be able to operate on different operating systems and devices
- The system should be able to work in an offline situation
- Each approval request should be processed within 10 seconds.

5.5 Business Rule

- The system should notify intended users immediately after the available quantity of the ingredients goes below 10 units

6. Other Requirements

What came up because of merging your requirements into the SRS template? How did using the SRS template change the way that you think about requirements? If you were to continue gathering more requirements, where would you focus your effort? Explain.

We collectively noticed that we repeated some requirements in our list. We also noticed we are missing some requirements that we could have added. Using the SRS template made the requirements more clear and readable especially when we categorized them under specific tasks. This way, we can now figure out what requirements we already covered and what requirements we did not cover. If we were to continue gathering more requirements, we would focus more on the logging in/out category since we felt like there could be more added to it. No one except one person wrote requirements on it.

Team 9

Daniel Wang, Maria Zhang, Trinh To, Phuong Vu, Safwaan Taher

Part #2: User Stories

User stories list:

1. As a BBB customer, I want to be able to view an accurate, full menu of the food truck. (Story)
2. As a BBB customer, I want to be able to use the application on my smartphone. (Saga)
3. As a BBB customer, I want the application to be responsive and fast. (Saga)
4. As a customer, I want my payment to be secure, so that I do not have to worry about my credit card information being stolen. (Saga)
5. As a customer, I want to know the business hours, so that I know when to order food. (Story)
6. As a customer, I want to have an option to list my allergies, so that I know my food would be safe to eat. (Story)
7. As the customer, I want to know if the system has any order limits, so that I can order one at a time if needed. (Story)
8. As the customer, I want to be notified if my order is almost ready, so that I can be prepared for when it will be done. (Story)
9. As a manager at BBB, I want to be able to edit and view work schedules. (Story)
10. As a manager, I want to get a report of sales and ingredients orders, so that I can learn whether the company is generating profits or not. (Epic)
11. As a customer services employee, I want to know the customer feedback of our food and services, so that I can send out notices for departments to make improvements. (Story)
12. As an accountant at BBB, I want to know the total sales report every 1 month, so that I can calculate the company expenses and employees commision. (Story)
13. As an event host, I want to customize my reservation of BBB's food trucks for my event, so that I can have the correct kind of food, time, and location, and avoid unwanted allergies. (Epic)
14. As a food truck cashier, I want the system to work well under peak hours with many orders. (Saga)

15. As a food truck worker, I want to be able to order new supplies when they get low. (Story)
16. As a food truck chef, I want to be able to add new dishes to the application. (Story)
17. As a food truck worker/chef, I want to be able to edit the menu in case items are out of stock. (Story)
18. As a food truck worker, I want to be able to view new, current, and past orders. (Story)
19. As the chef, I want to know if the customers have any special requests, so that I can exclude it from their order. (Story)
20. As a cook, I want to have access to the food truck's pre-order list, so that I can prepare the orders beforehand. (Story)
21. As a cashier, I want to send the order to the kitchen as soon as possible, so that I can increase the customer satisfaction. (Story)
22. As a food truck driver, I want to get an accurate and fast navigation route to the food truck's destination, so that I can drive to the destination safely and efficiently. (Story/Epic)
23. As a warehouse worker, I want to check for when the next delivery is scheduled, so that I can start to prepare for storage and receiving the delivery. (Story)
24. As a warehouse manager, I want to look for the delivery dates and the producing company of these boxes of forks stored in section 1 of the warehouse, so that I can send a complaint to the suppliers about the quality. (Epic)
25. As a warehouse worker, I want to have access to suppliers' inventory, so that I can place ingredient orders for the company's food trucks. (Epic)
26. As a BBB's Supplier, I want to know when a restock order has been placed and the quantities and types details, so that I can have my delivery on time and with as less error as possible. (Epic)
27. As a BBB's Supplier, I want to notify the BBB's Warehouse on the possible goods shortage, so that I can propose an alternative solution or types with a reasonable price. (Story)

INVEST rubric

	Qualified	Non-Qualified
Independent	(1-27)	
Negotiable	(1-27)	
Valuable	(1-27)	
Estimate	(1-27)	
Small	(1-27)	
Testable	(1-27)	

Part 3: Compare your SRS requirements and user stories

1. Internet of Things (IoT) application requiring custom hardware / software

   User story – IoT application needs the customer's input and perspective throughout the development process since it requires custom hardware/software. It is easier to meet customer's expectations and satisfy their needs with user stories because this process is more personable and conversational, rather than purely on procedural writing.

2. Offline / online data synchronization manager

   SRS requirements - Offline/ online data synchronization would require an established consistency among data from a source to a target data storage and vice versa. As a result, the manager would need a plan or prescriptive approaches for this system. SRS requirements would be ideal for this subsystem since the wordings and terms on it are consistent and agreed upon a set of terms that reflect the importance of the requirement.

3. Financial payment system that must meet PCI compliance

   SRS requirements - Since we have a PCI compliance requirement, we have requirements to follow so a SRS will be a safe options for the developing process. However, the update and maintenance process can be agile for flexibility.

4. Mobile application that focuses on information deliver

User Story - Mobile application information delivery works closely with end-users. It would be best to work with users to understand how they want the information and what information should be delivered. Making the information easy to read and understand is very important. Thus working with users and user stories will help with the development.

Part 4: Compare SRS and User Stories to Scenarios and Use Case Diagrams

In this part, consider how all of the artifacts / models you have created so far (personas, SPICIER, use case diagrams, SRS requirements, user stories) are different expressions of requirements / specifications. Examine all of these artifacts / models relate to each other, map to each other, overlap, or are distinct from each other. In what ways do the different artifacts / models support each other for the sake of defining the entire set of needs related to software development?

The overarching goal of all the artifacts and models is to get the stakeholders to think from a customer point of view. User scenarios outline typical technological frustration in a users life and personas detail what a customer or end users day to day life is. User stories and use case diagrams help create function and non-functional requirements by highlighting how a customer interacts with our system. Finally SRS organizes all the above concerns into potential features in functional and nonfunctional requirements.

In effect all artifacts help support each other. A persona can help a stakeholder understand a user and create a fitting user story, which in turn spawns requirements for the SRS. Use cases wrap this up by showing interactions between the system from multiple artifacts and models. In a sort of building block, akin to assembling a LEGO set given an instruction manual diagram etc, the artifacts/models each support each other in the end goal of understanding the customer and defining their needs.

To what degree are each of these artifacts / models inward-facing versus outward-facing from the perspective of the team creating these artifacts / models? Which of the artifacts /

models are more likely to *drive* the product development process, versus being *inputs* to the drivers? Explain.

The SRS is likely to drive the development process as it outlines concrete requirements which are testable and implementable. Another artifact that can drive the process is user stories as these outline specific uses of the system and can be an immediate catalyst for requirements or new features. As for each artifact inward/outward facing degree:

- SRS: About 50/50 to both team and customer
- User Scenarios: Majority outward facing from the team
- Use Case: Also a majority outward facing artifact from the team
- User Story: Nearly an even split still inward facing to a decent degree for the team.

Software developers seldom create and maintain these artifacts / models in complete and updated forms. Discuss and come up with what you think would be good guidelines about how to decide which artifacts / models to create, to what level of "doneness", update, distribute, etc.?

The most complete guideline to decide what artifacts/models to create is that any feature planned in the development lifecycle should be complete or at least 80% "done". This can ensure that features meet stakeholder needs as well as end user needs, a clear positive of the modeling/artifacting process. Use Cases and user stories should be made for each new feature, the older stories/cases don't need to be maintained they can just be overridden. In this way developers have the flexibility to work in a fast paced environment and grow the project while maintaining a level of compliance with artifacting/modeling.