VILNIUS UNIVERSITY

MATHEMATICS AND INFORMATICS FACULTY

SOFTWARE ENGINEERING

Locals to Locals

Third Laboratory Work

Software Engineering II

Submitted by:

Arnas Rimkus,

Deividas Kučinskas,

Matas Lazdauskas,

Jaroslav Kochanovskis,

Gytis Bečalis

Supervisor: Assist. Dr. Vytautas Valaitis

Vilnius 2021

Summary

As we continue to learn software engineering principles and apply them in development of an actual application, in this paper we will use the ICONIX process to analyze a change request in the system.

1. Choose a change request for the system.
2. Perform impact analysis, define implementation alternatives, estimates.
3. Define a project plan for implementing the change.
4. Elaborate the requirements using the ICONIX process.

This work will contain the change request impact analysis, project plan and the requirements.

Table of Contents

1. Context

After fulfilling the previous requirements, we have received a change request from the project owners. In this part, we analyze the change request and how it will impact the current system, while also examining the possible alternatives. Finally, we formalize the requirements of the change request and create a project plan to implement the change.

1.1 Change Request

The proposed change request is to create a swipe card page for buyers to find vendors, design displayed in Figure 1. This subsystem allows buyers to find vendors in a new way - vendor's service will be displayed one at a time, the order of which will be determined by distance. Also, the statistics of usage are to be tracked, so that vendors know how often they are skipped or selected. To denote popularity, trophies are to be displayed in the service page based and the card's border color is to be based on the select to skip ratio.

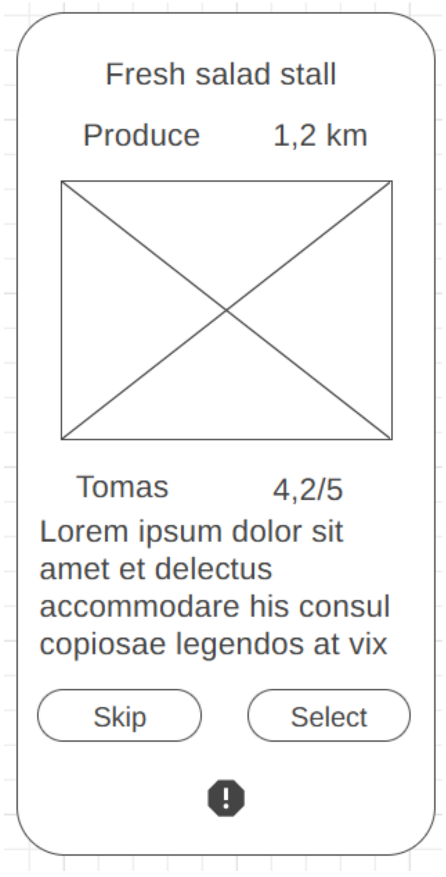

Fresh salad stall
Produce
1,2 km
Tomas
4,2/5
Lorem ipsum dolor sit amet et delectus accommodare his consul copiosae legendos at vix
Skip
Select

Figure 1. Swipe Card Design

1.2 Current Situation

As the main goal of the system is to connect buyers with vendors, having efficient and user-friendly methods to find services is crucial. Currently, it is possible to find services by:

1. Browsing the list of services, aided by filters
2. Browsing the map
3. Searching the services by name

While these methods work for desktop-based users, they are not user friendly for mobile users, even with the website employing a responsive design. As such, we believe, and the project owners believe that the system needs a method of finding services that is specifically designed with mobile users first in mind.

Additionally, no usage statistics are currently being tracked, which only allows vendors to measure their popularity based on the reviews received, which could make it difficult to judge whether our system is helping their business.

1.3 Change Request Impact Analysis

The proposed change, if implemented, would address several issues currently experienced by the system. First, it would add a method of finding services created specifically for mobile users first, which as mentioned before would be very beneficial to the system, both as an expansion of possible user base, and because that most users would likely want to use our system while outside their home. Additionally, we believe that the swipe card design is a more engaging activity for users, creating more exposure for the services. However, while the swipe card design would be more comfortable for mobile users, it is important to note that it is still not as effective as an app designed specifically for mobile.

Another positive impact would be introduction of statistics tracking, which would be beneficial both to the vendors and to the buyers. For the vendors, they would be able to monitor their services more efficiently, while the buyers would have additional information about the popularity of the service, which would help them choose the best service possible.

1.4 Implementation Alternatives

Before choosing the solution described earlier, we have analyzed several alternatives.

1.4.1 Alternative Solution - Mobile App

The first alternative we looked at was the creation of a mobile app in addition to our web-application. This would allow us to create a platform specifically designed for mobile users, significantly improving our reach in this user segment. However, the cost of this change would be too great, as it would require creation of an app from scratch, while our development team is small, and we are still in the middle of development of the web-application.

1.4.2 Alternative Solution - Partial Implementation

Another alternative that was considered is implementing only a single part of the change request - either statistics tracking without swipe card page or reverse. This alternative would allow us faster and simpler implementation, while leaving the ability to implement the other missing part later. However, we believe that we have enough manpower in our team to develop both parts of the change request concurrently and as such there is no need to split the development.

1.4.3 Alternative Solution - No Changes

Finally, we examined the solution to not implement any changes regarding the current issues faced. While this would be the cheapest and least time-consuming option, allowing us to focus on other ideas to improve the system, we believe that the issues we have found should have priority against other possible avenues of improvement.

1.5 Requirements

To clarify the exact changes to be made, we have created a formalized list of requirements. To note, requirements are usually classified as functional and non-functional, however in accordance with recent research we have chosen the name quality requirements instead of non-functional as to not understate the importance of these types of requirements.

1.5.1 Functional Requirements

FR 1. Create a swipe card page for buyers to find vendors' services

- FR 1.1. Page must contain a swipe card with:
  - FR 1.1.1. Name of the service

FR 1.1.2. Image of the service

FR 1.1.3. Type of the service

FR 1.1.4. Name of the vendor

FR 1.1.5. Distance to the service

FR 1.1.6. Service average review score

FR 1.1.7. Short description

FR 1.1.8. Button to select service, which redirects to vendors service details page

FR 1.1.9. Button to skip service

FR 1.1.10. Button to report service

FR 1.1.11. The card border color is to be determined by the card’s score.

FR 1.2. Swipe cards to be arranged in order of distance from the buyer

FR 1.3. If all services are looked at, start again from the beginning of the list

FR 2. Track statistics:

FR 2.1. The system should track the number of times the service was viewed, selected, and skipped.

FR 2.2. Display statistics to the vendor.

FR 2.3. Add visual trophies to the vendor's service to display popularity.

1.5.2 Quality Requirements

QR 1. Usability requirements

QR 1.1. All updates must be in English and Lithuanian

QR 1.2. All updates to be designed with mobile UI in mind first

QR 2. Capacity requirements

QR 2.1. New features should not impact the general performance of the website (average load times to remain in the same 10% timeframe)

QR 3. Availability requirements

QR 3.1. New features must work just as well as the whole service

QR 4. Interoperability requirements

QR 4.1. Usage of PostgreSQL

QR 5. Reliability requirements

QR 5.1. New features should be relatively bug free (no unhandled crashes)

QR 6. Regulatory requirements

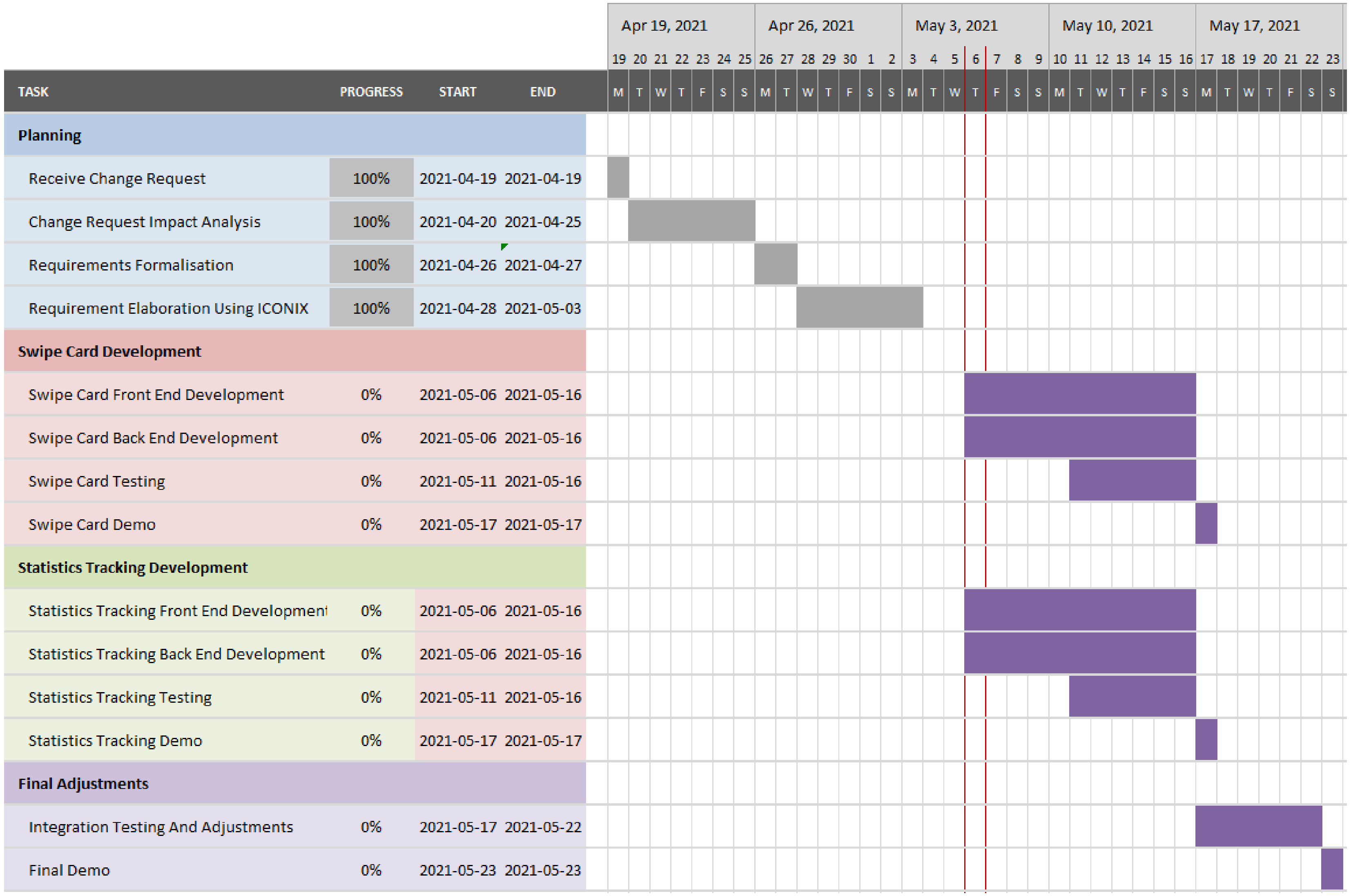
QR 6.1. Comply with GDPR requirements

1.6 Project Plan

To efficiently plan the project timeline, we used the Critical Path Point method. The resulting table is shown in Table 1. With this information, we created a Gantt diagram, shown in ssFigure 2, to display projected timeline. We have already marked the planning process as complete in the Gantt diagram, and if the project is to continue as planned, after evaluation of this work we would proceed with the development of the changes discussed in this document.

Table 1. Critical Point Path

Key	Name	Prerequisites	Length (Days)	Early Start	Early Finish	Late Start	Late Finish
A	Receive Change Request		1	0	1	0	1
B	Change Request Impact Analysis	A	5	1	6	1	6
C	Requirements Formalization	B	2	6	8	6	8
D	Requirement Elaboration Using ICONIX	C	5	8	13	8	13
E	Swipe Card Front End Development	D	10	13	23	13	23
F	Swipe Card Back End Development	D	10	13	23	13	23
G	Swipe Card Testing	D	5	13	18	18	23
H	Swipe Card Demo	E, F, G	1	23	24	23	24
I	Statistic Tracking Front End Development	D	10	13	23	13	23
J	Statistic Tracking Back End Development	D	10	13	23	13	23
K	Statistic Tracking Testing	D	5	13	18	18	23
L	Statistic Tracking Demo	I, J, K	1	23	24	23	24
M	Integration Testing and Adjustments	H, L	5	24	29	24	29
N	Final Demo	M	1	29	30	29	30



TASK	PROGRESS	START	END
Planning			
Receive Change Request	100%	2021-04-19	2021-04-19
Change Request Impact Analysis	100%	2021-04-20	2021-04-25
Requirements Formalisation	100%	2021-04-26	2021-04-27
Requirement Elaboration Using ICONIX	100%	2021-04-28	2021-05-03
Swipe Card Development			
Swipe Card Front End Development	0%	2021-05-06	2021-05-16
Swipe Card Back End Development	0%	2021-05-06	2021-05-16
Swipe Card Testing	0%	2021-05-11	2021-05-16
Swipe Card Demo	0%	2021-05-17	2021-05-17
Statistics Tracking Development			
Statistics Tracking Front End Development	0%	2021-05-06	2021-05-16
Statistics Tracking Back End Development	0%	2021-05-06	2021-05-16
Statistics Tracking Testing	0%	2021-05-11	2021-05-16
Statistics Tracking Demo	0%	2021-05-17	2021-05-17
Final Adjustments			
Integration Testing And Adjustments	0%	2021-05-17	2021-05-22
Final Demo	0%	2021-05-23	2021-05-23

Figure 2. Gantt Diagram

2. Static Model

As part of the ICONIX process, we have developed a static model for the proposed change request implementation. Our main goal here is to develop the initial domain model, as well as to visualize a class diagram for the changes, which will help define the use cases that will be following later.

2.1 Domain Model

Each service has a single swipe card. On the swipe card, its service details are displayed: its name, image, type, name of the vendor, distance to it, average review score and a short description. Vendors who own a service can check its statistics in the corresponding service page. These statistics include the number of times the service was viewed, skipped, and selected. They are tracked via user input in the swipe card page. Every service permanently displays a trophy visible in its service page and a special border visible in its swipe card when certain requirements, such as times the service is viewed, are met.

The domain model is visualized in Figure 3.

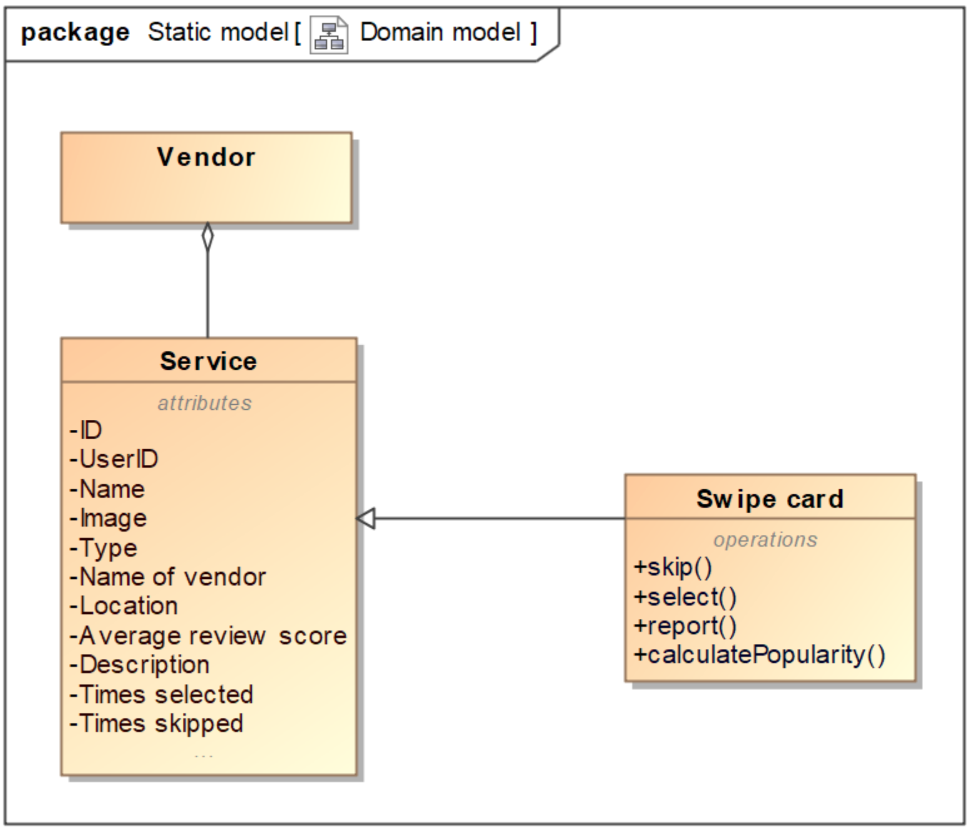


Figure 3. Domain Diagram

2.2 Class Diagram

To integrate this swipe card addition, we must take the service information, which belongs to the “Vendor” class, process it, and then display the results to the user. We are planning to achieve this with the MVC model - with the “SwipeCardModel” class for keeping track of service data, the “SwipeCardController” class for doing the needed operations and the “SwipeCardView” class for working with the UI. The class “SwipeCardModel” is required because a swipe card has information about the service it is displaying along with new data to keep track of and work with. The swipe cards are also dependent on the services they are displaying and cannot exist without them. Operations are mainly required to ascertain the distance to each service to figure out the order of displaying, and to award a trophy to them, when it is earned via specific user input. This user input is as follows: the amount of time users selected to view the service, the amount of time users chose to skip the service and view a new one, the number of reports, if any.

The class diagram is shown in Figure 4.

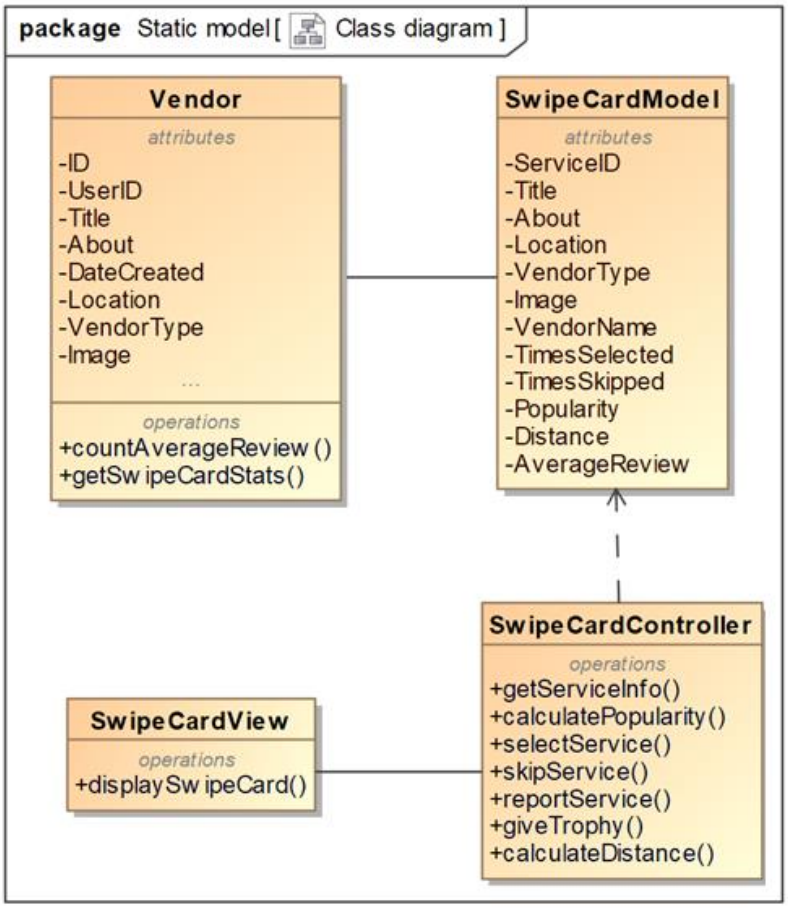


Figure 4. Class Diagram

3. Dynamic Model

We identified how the user will interact with the system using text and use case diagrams. Use cases provide a structured way to capture not only “sunny-day scenarios”, but “rainy-day scenarios” as well.

3.1 Swipe Card Use Case

Main scenario: The buyer chooses to open the Swipe card page by clicking the button in the navbar and the system opens the page, and the nearest unshown service card is displayed. After acquainting with the info provided in the card the buyer either chooses to skip the service or view the currently displayed service by clicking the Select button. If the Select button is clicked the buyer is then taken to the service page.

Alt. 1: **If the skip button is clicked** another nearest unshown service is displayed and the cycle repeats.

Alt. 2: **If there are no unshown services left** the list starts again from the beginning.

Alt. 3: **If the buyer is not logged in** then the buyer is first taken to the Login screen and then to the Swipe card page once he is logged in.

Alt. 4: **If the information in the service card is inappropriate,** a report form is provided and after User submits a report another nearest unshown service is displayed and the cycle repeats.

These scenarios are visualized in Figure 5.

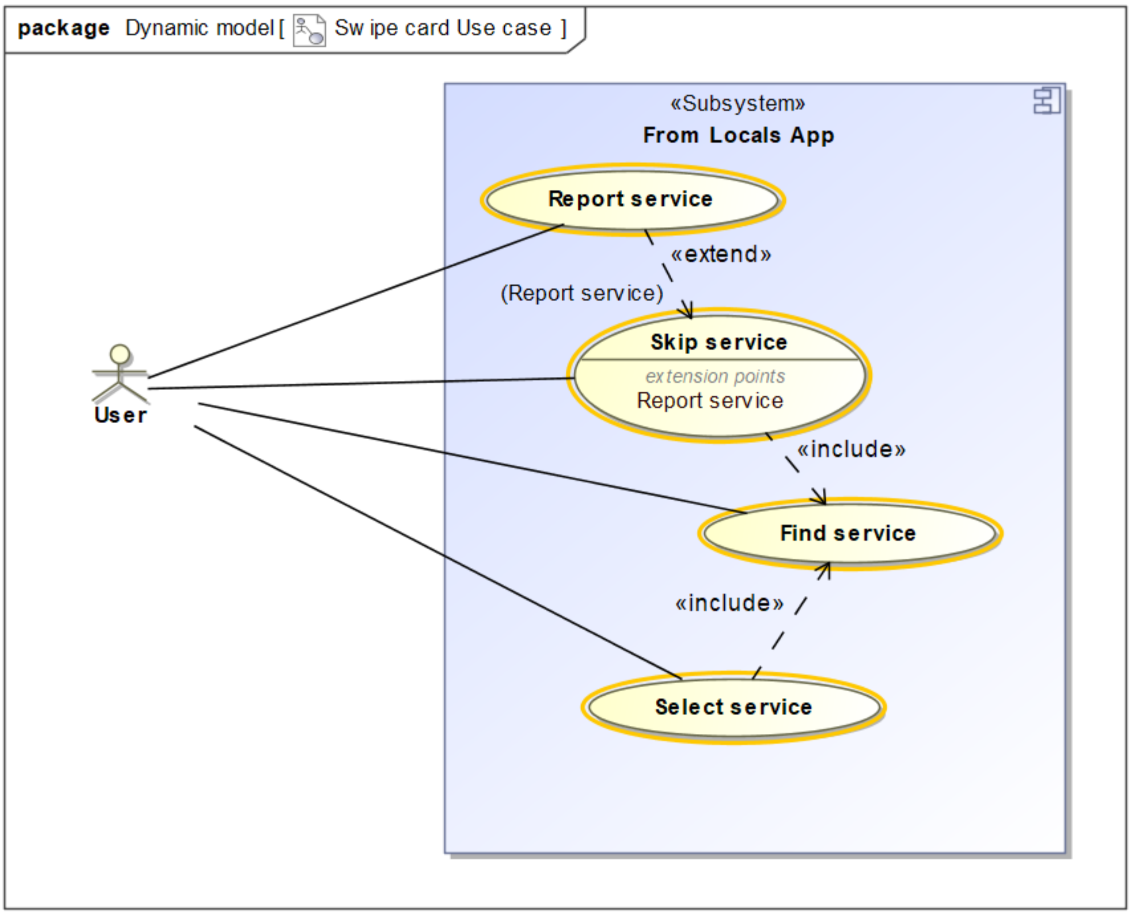


Figure 5. Swipe Card Use Cases

3.2 Popularity Use Case

Main scenario: In the service page users can see visual trophies score which the System calculates by tracking the number of times the service was viewed and selected by the Users in the Swipe card page. By using this information User determines the service's popularity.

Alt. 1 **If the service's swipe card has never been skipped or selected by a User** score is not calculated and displayed.

These scenarios are displayed in Figure 6.

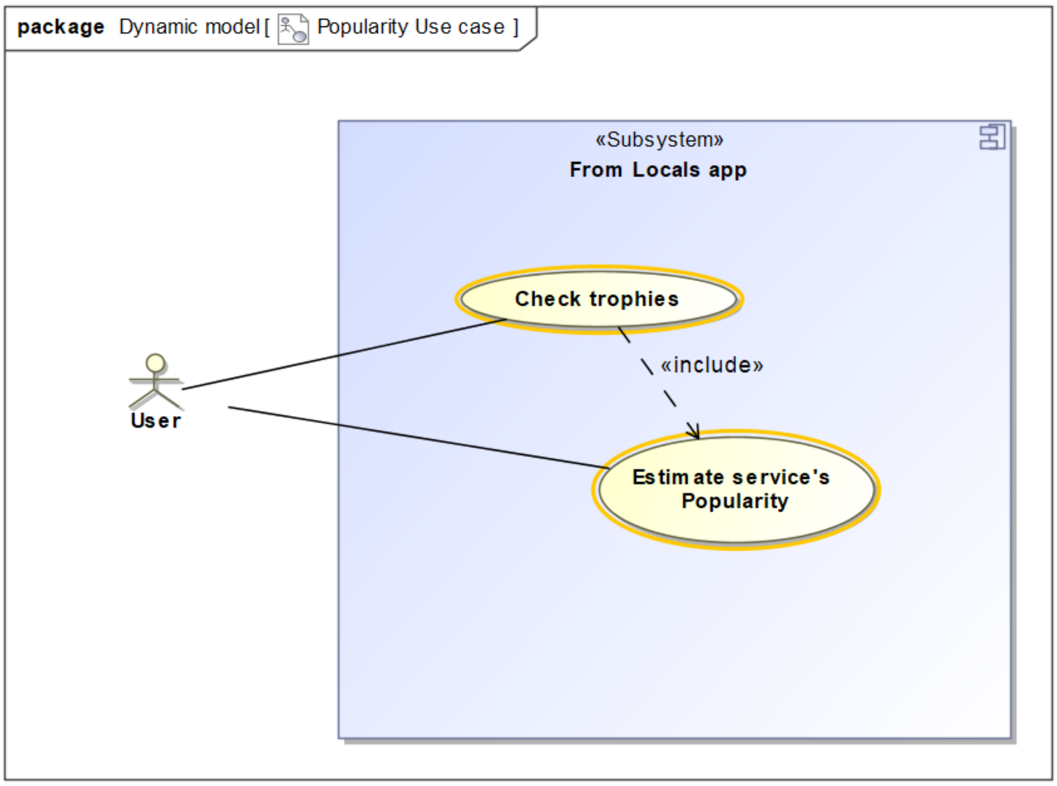
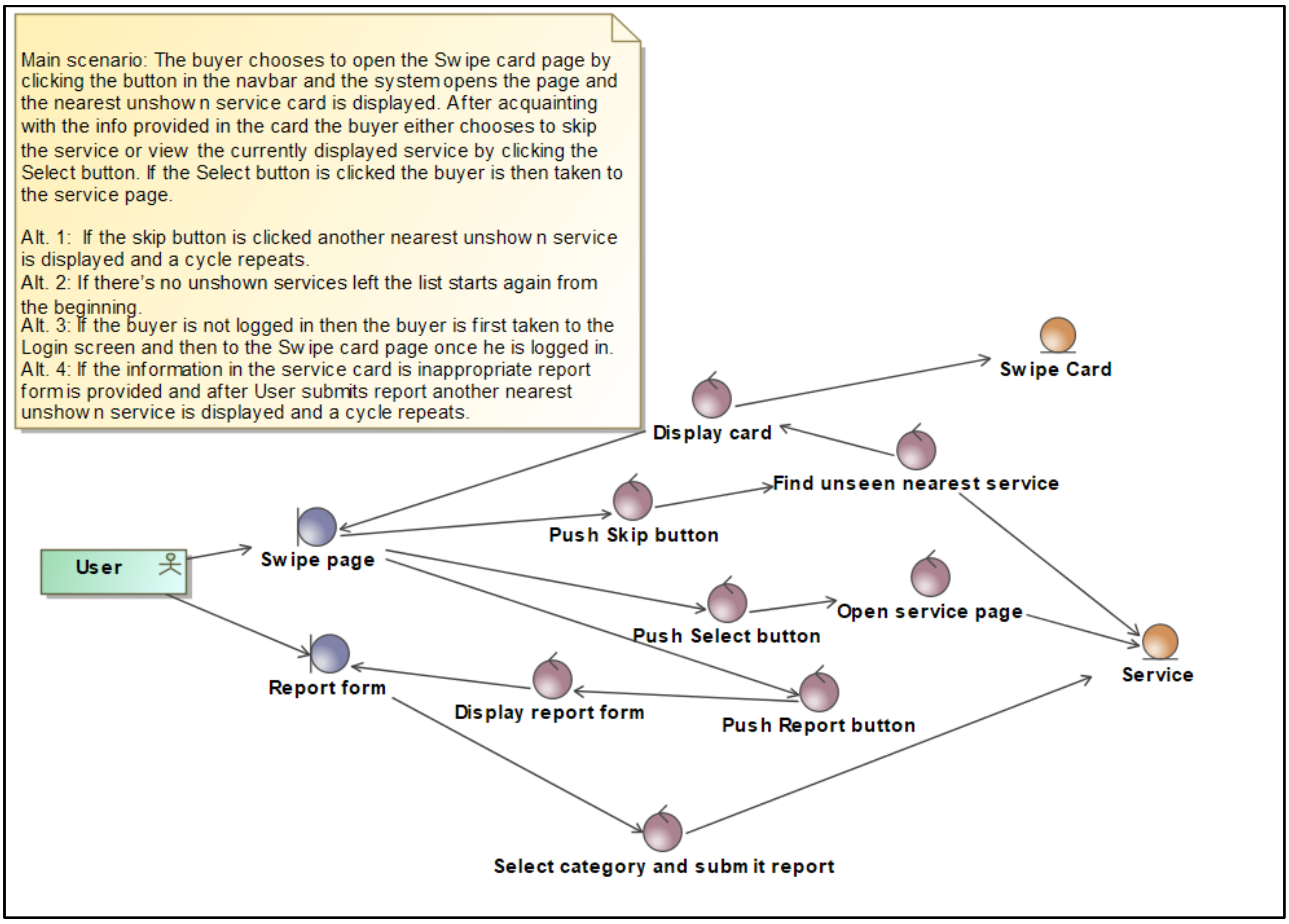

package Dynamic model [Popularity Use case]
«Subsystem»
From Locals app
Check trophies
«include»
User
Estimate service's Popularity

Figure 6. Popularity Use Cases

3.3 Swipe Card Robustness Diagram

To link use cases to objects and bridge the gap from analysis to design we created a robustness diagram, shown in Figure 7. This part helps to make preliminary assumptions about the design of the requested change.


Main scenario: The buyer chooses to open the Swipe card page by clicking the button in the navbar and the system opens the page and the nearest unshown service card is displayed. After acquainting with the info provided in the card the buyer either chooses to skip the service or view the currently displayed service by clicking the Select button. If the Select button is clicked the buyer is then taken to the service page.

Alt. 1: If the skip button is clicked another nearest unshown service is displayed and a cycle repeats.
Alt. 2: If there's no unshown services left the list starts again from the beginning.
Alt. 3: If the buyer is not logged in then the buyer is first taken to the Login screen and then to the Swipe card page once he is logged in.
Alt. 4: If the information in the service card is inappropriate report form is provided and after User submits report another nearest unshown service is displayed and a cycle repeats.

Figure 7. Robustness Diagram

4. Model to Functional Requirements Traceability Matrix

We created a model to functional requirements traceability matrix, shown in Table 2, to ensure that all requirements are covered in our requirements elaboration. We skipped requirements FR 1.1.1 – FR 1.1.11 since they strongly relate to FR1.1 and no model/diagram addresses them separately.

Table 2. Model to Functional Requirements Traceability Matrix

	FR	FR 1.1.	FR 1.2.	FR 1.3.	FR 2.1.	FR 2.2.	FR 2.3.
Model							
Domain Model		X			X	X	X
Class Diagram		X			X	X	X
Swipe Card Use Case		X	X	X			
Popularity Use Case					X	X	X
Swipe Card Robustness		X	X	X			

5. Results

In this laboratory work, we have achieved the following results:

1. Performed impact analysis by analyzing the current situation of the system, the probable effects of the proposed change and inspecting the possible alternatives to the change request
2. Formalized the requirements
3. Created a project plan using the Critical Path Point method and created a schedule using a Gantt diagram
4. Elaborated the requirements using ICONIX

6. Conclusions

From the results of this laboratory work, we can draw the following conclusions:

1. The proposed change would be beneficial to the system, especially to mobile users
2. We have chosen to implement the proposed change against other alternatives, as we deemed it the most beneficial
3. The proposed change is feasible to implement in the required timeframe with our current resources