ICT Engineering

**SEP1Y**

**Process Report**

Submitted to:

Steffen Vissing Andersen

Birgitte von Fyren Balsløv

Prepared by:

Sebastian-Dan Basca

Martin Kostadinov

Octavian Grozman

Mihai Timotin

Submitted on : INSERT DATE HERE



**Contents**

1. Group policy

2. SWOT Analysis

2.1. Group SWOT Analysis

2.2. Individual SWOT Analysis

3. Considerations

4. Daily Project Log

5. Task Log

6. Meeting

7. Bloom’ forms

**Group Policy**

The following group rules were established and approved by the entire group before starting working on the semester project.

* Always contact all group members when we are going to miss a meeting.
* Work on group assignments collaboratively and cohesively.
* Switch roles for each assignment.
* Show up on time at meetings.
* All members are notified about the future meetings within at least 12 hour notice.
* Complete the assignments during group meetings (in case we do not manage to complete the tasks during the meetings, we should finish everything at home).
* Assist other members when they encounter problems with their assignment.
* Show interest and passion for the project and be serious as much as possible.
* Communicate in English only.
* Communication between members can be done through Facebook group and also on Skype.
* Take into consideration everyone’s opinion.
* Be open-minded to new suggestions.
* Prioritize SEP over other activities in your schedule.
* Do not disrespect other members of the group, try to behave professionally.
* There will be a weekly discussion about the progress of the project.
* All members will have something to work on.

**SWOT Analysis**

1. Group SWOT Analysis:

**Strengths:** Enthusiasm, positive group members, motivation to succeed, hard-working, good team chemistry, open-minded members

**Weaknesses**: Little experience with group projects, many contradicting opinions, limited creativity, easily distracted.

**Opportunities**: Improve programming and team working skills (proper communication), learn to take responsibility, learn to pay more attention to details.

**Threats**: Splitting the amount of work properly, time management issues (setting up team meetings).

1. Individual SWOT Analysis:

**Martin Kostadinov:**

**Strengths**: Motivated to succeed, positive behavior, practical, reliable.

**Weaknesses**: Easily stressed, selfish, worrying about small details.

**Opportunities**: Develop teamworking and social skills, learn more about programming, get a lot of practical experience.

**Threats**: No specific threats.

**Mihai Timotin:**

**Strengths**: Punctual, creative, optimistic.

**Weaknesses**: Easily distracted and bored.

**Opportunities**: Getting experience in group programming, expand IT knowledge, learning to work in a team environment.

**Threats**: No specific threats.

**Octavian Grozman:**

**Strengths**: Experienced in programming, good communication skills, cooperative, friendly, resourceful and creative.

**Weaknesses**: Not very punctual, inflexible, not very efficient with time spending.

**Opportunities**: Improve programming and communication skills, become better organized, learn new things, improve presentation skills.

**Threats**: No specific threats.

**Sebastian Basca:**

**Strengths**: Reliable, passionate, hard-working.

**Weaknesses**: Laziness, postponing work until last moment.

**Opportunities**: Improve group work and IT skills, gain wider understanding of working with other people.

**Threats**: No specific threats.

**Considerations**

Sebastian-Dan Basca

* Add essay

Martin Kostadinov

When the project period started, I was very excited about this new challenge and was highly motivated to succeed. It really helped that everyone in our group shared the desire to make the project as good as possible. As we started, I was confident in our group, but I was aware of our inexperience. However, everyone remained positive which was important for our team mentality. In the very first week, we were meeting a lot and making sure that we are all one the same page and if we weren’t – we would discuss our ideas and choose the optimal path. Everyone was open-minded and we were all working towards the same goal. As time went by, we continued to meet on a regular basis and we often split the workload as to provide maximum efficiency. Very often we also went over the work we had done so far where everyone had the chance to explain how they approached their tasks. This way of keeping everyone updated about everything was very important to us, because we were all so eager to learn and felt that the everyone should know how all the parts of the project were made. And even if it meant spending more time talking about their particular tasks, it was very much worth it. Since this was our first group project, we were making a lot of mistakes, which was humbling and helped us learn more about the process of software development. The middle and the end of the project period was with most challenges, as this is when we had to do most of the work. However, with a lot of cooperation by everyone, and clear communication, we were able complete the project. I was very happy with the group I chose, because everyone was working hard and making sure we all succeed. I learned a lot during this first project period which I’m sure will be helpful in the future and I can gladly say it was a very educating experience.

Octavian Grozman

* Add essay

Mihai Timotin

* Add essay

|  |  |
| --- | --- |
| **Daily Project Log** | |
| **PROJECT NAME** | **SEP – VIA BUS** |
| **GROUP MEMBERS** | Octavian Grozman, Martin Kostadinov, Sebastian Basca, Mihai Timotin |
| **DATE** | **TASK** |
| 11 October | For the first we met in order to create and sign the group policy. |
| 5 November | Discussion of the interview, determining the main points about the future system. |
| 7 November | Writing down some requirements |
| 10 November | Continuing to write down the requirements for the system |
| 12 November | General discussion about the system and fixing the requirements. |
| 15 November | Starting work on Use Case Diagrams |
| 17 November | Working on Use Case Diagrams and also Use Case Description |
| 20 November | Finishing the Use Case diagrams and descriptions, and starting work on Activity Diagrams |
| 23 November | Working on Activity Diagrams, fixing the Use Case elements |
| 25 November | Starting work with Class diagram, working on Activity Diagrams |
| 27 November | Finishing the Activity diagrams, planning the model classes. |
| 29 November | Group discussion about the project structure: MVC components, building up the model, File system. |
| 1 December | Finishing the Class Diagram (Condensed version) |
| 2 December | Complex programming on creating the base model classes and Lists classes. |
| 5 December | Creating the File System and connecting it with Model. |
| 6 December | Making the complex Class diagram and fixing the bugs with file system. |
| 7 December | Group discussion about GUI design and drawing the sketch for the system. |
| 8 December | Splitting up the work on doing the User Guide, Bloom forms, UML, Sequence diagram, SWOT analysis and updating the old Introduction with Group Policy. |
| 9 December | Programming together on GUI elements |
| 10 December | Fixing some problems with GUI, creating the main GUI elements (lists, tabs, buttons, labels, textfields) |
| 11 December | Finishing the GUI (creating and binding of event listeners) Connecting the model with GUI properly. |
| 12 December | Making various tests together and checking the system functionality, fixing the found bugs. |
| 13 December | Final testing of program and creating the layout for Documentation. System is ready. |
| 14 December | Writing the bloom forms, condensation essays, SWOT analysis, personal process reports and writing the documentation. |

Group task log

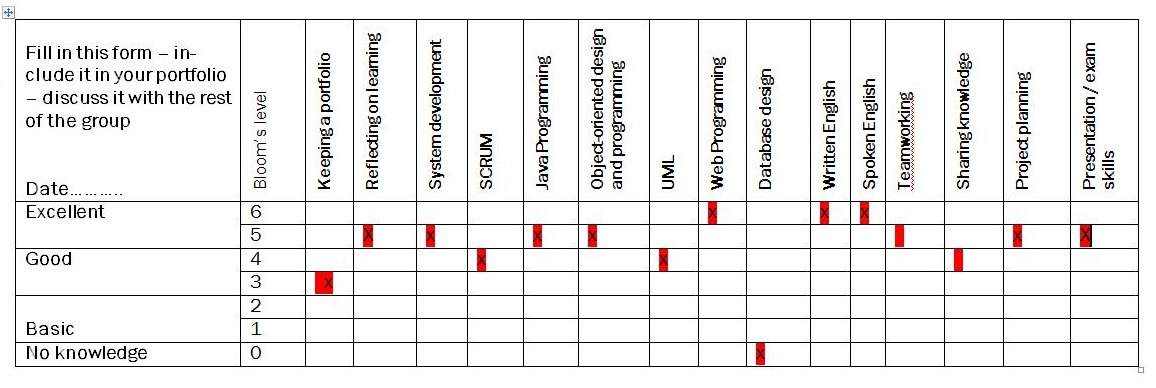
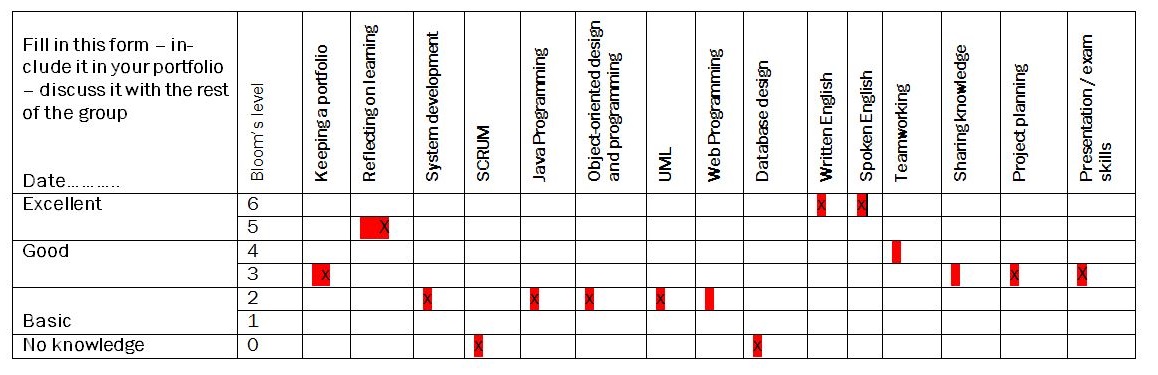
Octavian

**Bloom’s Forms**

**Sebastian-Dan Basca:**



**Martin Kostadinov:**



**Octavian Grozman:**



**Mihai Timotin:**

****

ICT Engineering

**SEP1Y**

**Project Report**

Submitted to:

Steffen Vissing Andersen

Birgitte von Fyren Balsløv

Prepared by:

Sebastian-Dan Basca

Martin Kostadinov

Octavian Grozman

Mihai Timotin

Submitted on : INSERT DATE HERE



**Contents**

1. Abstract

2. Introduction

3. System Analysis

3.1. System Requirements

3.2. Priority Use Case Diagram

3.3. Use Case Description – Manage reservations data

3.4. Activity Diagram – Manage reservations data

4. Design 4.1. UML Class Diagram (Condensed)

4.2. UML Class Diagram (Expanded)

4.3. Sequence Diagram – EDIT THIS

5. Implementation

6. Test

7. Results

8. Discussion

9. Conclusion

10. Reference

**Abstract**

*The reason we are creating a Single User System is to help a company that deals with a large amount of data, as well as a lot of customers, to be able to modernize and automate a lot of their daily tasks. The idea is to create a simple and useful application, for the company VIA Bus, that will be capable of dealing with the adding/editing/removing of reservations, tours, chauffeurs, customers and passengers. The approach we took rewarded simplicity and straightforwardness. By making our GUI as user-friendly as possible, we were making sure that intensive training wasn’t required. Our code is well structured and easy to maintain if needed. We’ve made a very easy to get into system that even though lacks immense depth, it still handles all the necessary requirements well. Overall, the system is not something revolutionary, but instead our approach of taking several tasks and putting them in an environment where they are easy to perform and can save a lot of manual work.*

**INTRODUCTION**

Our team has been given the task to develop a booking system for the company VIA Bus. An interview was conducted with the owners of the company in order to decide what were their requirements. It was decided that the system must allow employees to handle reservations for - seats for trips and journeys to different destinations or a whole bus with personal preferences (food, party guide, additional stops) to desired destination. The system contains information about fixed tours(trips and journeys),and non-fixed tours(bus-and-chauffeurs) as well as chauffeurs, customers, and passengers data. The company decided that only employees will be able to access and work with the system(add/edit/remove tours, reservations, chauffeurs passenger and customer data).

The system has a simple user interface with interactive elements.  
It is developed using Java programming language.

The purpose of this system is to make it easier and more efficient for VIA Bus to manage all the reservations and keep track of all the data that the company works with. In the following pages of the report, we will go in greater detail about the things we’ve done to make all of this possible and create a simple, yet efficient system.

**System Analysis**

**Requirements**

1. An employee should be able to make a reservation for a customer (trips, journeys and Bus-and-Chauffeur).
2. Only employees of VIA BUS have access to the system.
3. An employee should be able to add food preferences (breakfast, lunch, snacks) for longer tours.
4. Journeys and Bus-and-Chauffeur services longer than one day can have 2 chauffeurs.
5. An employee can create a tour by specifying the kind of tour (one-day trip, journey and Bus-and-Chauffeur), departure date and time, return date and time, number of seats, select an available bus, select one or more available chauffeurs, destination(s), stops and price for one seat.
6. An employee should be able to register information about customers and all the passengers (only for trips and journeys).
7. An employee should include the customer’s or company’s name if it is a bus-chauffeur service.
8. Each customer’s information should contain: name, address, phone number and e-mail address if he or she wants to get newsletter and company name if applicable.
9. Each passenger should contain: name, age and address only for trips and journeys.
10. An employee should be able to delete a reservation.
11. An employee should be able to edit a reservation (if there are enough seats left on that tour), customer’s info, chauffeurs’ info and passengers’ info.
12. An employee should be able to access the available list of busses while making a reservation.
13. An employee should be able to see the Chauffeur’s profile.
14. Each Chauffeur’s profile contains : name, phone number, form of employment (part-time or full-time), preferences, ID, e-mail, availability.
15. An employee should be able to add a new Chauffeur.
16. A customer should be able to reserve a party guide for the bus-and-chauffeur services that have a party bus.
17. An employee should be able to add a new bus to the current list of busses.
18. Frequent customers can benefit of discounts.
19. The system should display if the customer made reservations in the past so he may get a discount from the employee when he will make a new reservation.
20. The system should have a list of tours (trips and journeys) with fixed destinations.

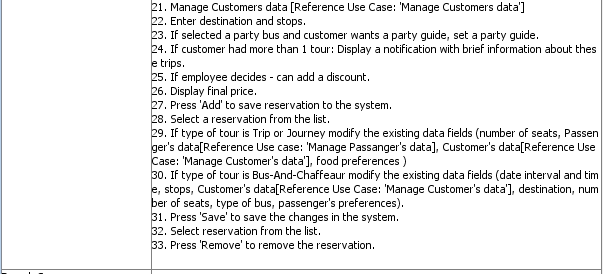
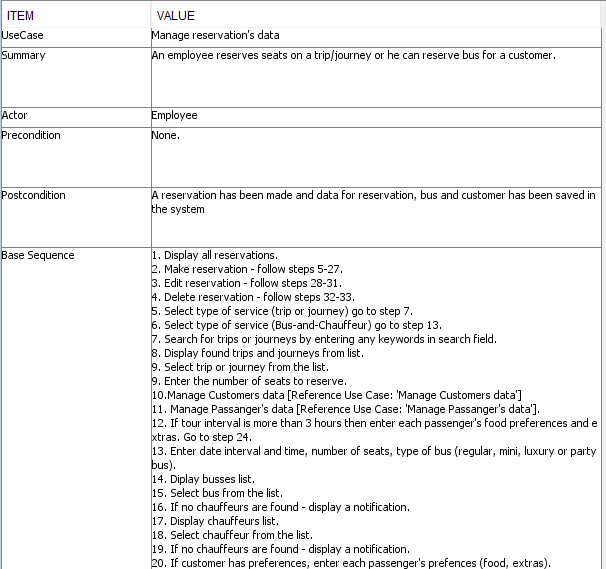
**Priority Use Case Diagram**

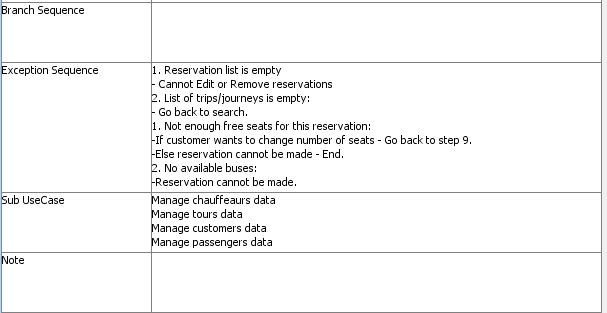


Priority Use Case Diagram

Illustrates the priority use cases performed by the employee of the bus company. It is the essential use cases required to add change or edit a tour chauffeurs or busses.

**Use Case Description – Manage Reservations Data**





**Activity Diagram – Manage Reservations Data**

**MIHAI YOUR JOB**

**UML Class Diagram (Condensed)**

Sebastian your job

**UML Class Diagram (Complex)**

**Breakdown**

**SEBASTIAN YOUR JOB**

**Sequence Diagram**

**OCTAVIAN YOUR JOB**

**Implementation**

**COMING SOON THE REST...**