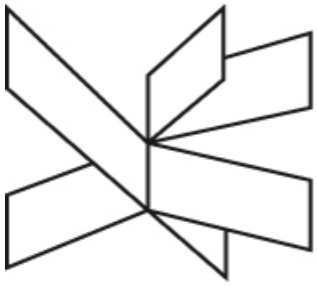


VIA University College

PROCESS REPORT

Semester1:

Software technology Engineering

Ameya Mahakal

Bozhidar Manev

Joan Tammo

Radoslav Kiryazov

Zsolt Nóvé

Character count: 24006.

Supervisors: Allan Henriksen, Mona Wendel Andersen.

Date: 01/06/2022

Table of contents:

1. Introduction:

The documenting of the project's process has given the group a chance to trace the progress of our group work from its formation until the handing of this document. Some of the trends we found were that the group follows a division of labour model of group work and the frequency of group meetings. This is well evidenced by the logbook which indicates the methodology by which we worked.

The frequency of the group's meeting and the division of labour is the model the group found most convenient for conducting group assignments because during the meetings the group would discuss the given assignment and put-up plans to advance with the assignment and divide the tasks equally.

The development process in which the project was conducted is an example of the waterfall model. It states that there are 4 stages in the development process, which are the following:

1. Analysis: in this stage, the group discusses the demands of the customer from which they deduct the system requirements. This was done in the case of the Overlook hotel using the interview which was conducted with the manager of the hotel.

   Then the group focuses on developing the domain model, creating use case diagrams, use case descriptions and activity diagrams.

   The analysis stage focuses on distinguishing the different requirements and developing plans accordingly. This facilitates the process for the next stage.

This process was done through the project description and analysis phases of the project in which the group focused on strengthening their understanding of the assignment.

2. Design: designing the program taking into consideration the analysis. The design stage focuses on the further development of the class diagram (the classes, the GUI, and the file handler), and the sequence diagrams.

The group made several iterations of the class diagram, starting with a proof of concept (a bare-bone class diagram), which bloomed into a more detailed and complex class diagram.

The design phase is about making a plan that the group can rely on, so the process of development of the program is less chaotic, thus avoiding misunderstandings.

3. Implementation: this stage focuses on implementing the designed UML diagrams. During this phase, the need to go back to the design phase may arise depending on the implementation process, because the Waterfall model allows us to go back to previous stages and make changes required by the implementation.

4. Testing: it is the final stage of the Waterfall model, in which the group is testing the product from the implementation. If required, the group returns to one or more previous stages to implement changes, resulting in a well-running product.

We conducted multiple tests that resulted in the discovery of shortcomings of the program, and thus going back to previous stages was needed.

The testing phase is heavily required since it evaluates the functionality of the program, leading to better-developed programs.

The Waterfall model and the division of labour are the best-suited methods for our group due to the flexibility that these methods provide for the group to conduct their work. They allow for independent work for the members and also allow for the development of plans that help regulate group work.

2. Group description:

The group is composed of a variety of nationalities. The 5 members are:

- **Ameya Sundeep Mahankal**: A student from India with basic experience in Python and MySQL.

- **Bozhidar Ganchev Manev**: A student from Bulgaria, who moved to Denmark to make his first steps into Software Engineering. He has a little experience in Java before his university education.

- **Joan Tammo:** A Kurdish refugee from Syria. His expertise with software in general before the start of his education is non-existent. Software engineering is his first experience with the development of software.

- **Radoslav Kostov Kiryazov**: A student from Bulgaria, who moved to Denmark to pursue education in the field of Software. Before that, he has little background in C#.

- **Zsolt Nóvé**: I am from Hungary and lived in Denmark for 3 years. I did not have a lot of programming knowledge before the start of my studies at VIA University. My small knowledge comes from game programming.

3. Project Initiation:

The beginning phase of the project was the formation of the groups. The groups performed different exercises within themselves (such as evaluations of different theories regarding motivation, group dynamics, and the description of the project). These exercises helped form bonds within the groups and helped the members assess each other, and their position within the group.

The initiation phase also consisted of working on the UML diagrams for the project, and during the RDW1 sessions, the group was given the task of constructing a website that is related to the project. This gave us the chance to develop a method for the group's approach to group work, which was to divide the tasks among the group members and hold meetings regularly to assess the work of each member. This approach helped us to develop an understanding of each-others work individually, and thus all the members understood the mechanics of the entire project.

The goal of group formation and the exercise which we practiced during SEP1 sessions was to make the group members more familiar with each other, and more knowledgeable of their capacity of what they can provide for the group on an individual level. This process can be explained by the use of the Tuckman's Team And Group Development Model or Tuckman's Model in short.

The Tuckman's Model suggests that a group's performance develops from their forming to the end of their project in 4 phases which are:

1. Forming: in this phase, a collection of individuals come together and form a group, and since the members are unfamiliar with each other it is hard for them to work together.

   This was the case for our group, for we weren't familiar with one another, and thus there were some conflicts such as some group members did not have the experience with programming in general, and it took them some time to develop the required skills for the programming tasks.

2. Storming: in this phase, the different members are in an assessment phase, where each individual is figuring out their position in the group, and in this phase, many conflicts may arise, however in our case we did not come across any major conflicts because individually each member got a picture of their position within the group because the group avoided any form of

hierarchy, and thus every member was in an equal position compared to other members.

3. Norming: in this phase, all group members become familiar with each other and thus it becomes easier for them to work together properly. In our group's case, we came to this phase quickly and we were able to form an idea of how to approach a task which was by dividing the labor equally and helping group members who are struggling with their designated task.

4. Performing: in this phase, the group members become familiar with each other and can approach tasks equipped with knowledge of each individual's strength and thus can develop a plan of how to approach the tasks more properly, and in our case, we were given the chance to evaluate our group performance during the RDW1 assignments. In this assignment, each member took the responsibility of developing a section of the assignment and during our meeting, we evaluated each member's work and combined the different parts of the assignment to form the final product.

The different teaching opportunities that our group was presented with during the SEP1 sessions helped us in developing our approach to the task we were given, and thus the group was able to proceed with the task without any major struggles.

4. Project description:

During the project description phase, the group's approach was to analyze the material (an interview with a hotel manager) presented to us, and extract the different requirements from it, such as the clients' requirements for the program. Afterwards, the group divided the documentation process among the members where each member was responsible for a section. The goal of this was to analyze the material and present the knowledge the group gained during the analysis phase.

This approach is an implementation of deep learning for the analysis phase, and surface learning for the documentation phase.

In the analysis phase, the deep learning process is implemented because the purpose of analyzing the presented material was to develop a critical analysis of the requirements that help in the further development of the required program.

In the documentation phase, the surface level of the learning process is implemented because the goal of the documentation process is to present surface-level information to a target audience (the supervisors), and this requires a surface level understanding of the material to document such material.

The project description phase helped the members of the group in developing a proper method of documenting information regarding the project. This was done in two phases, the first one was a critical analysis of the presented material, and the second phase was documenting a surface level of information that helps in developing proper presentation skills for the target audience.

5. Project execution:

The project execution phase was an opportunity for the group to combine the knowledge gained throughout the semester. This project helped the group realize the importance of combining our experiences to achieve a better outcome, and this was well evidenced during our meetings, in which whenever the less experienced members would face struggles with their tasks a more experienced member would be there to help out. The management of our group was a responsibility that some members were better at, thus they were left to organize the individual tasks each member was to follow, and thus no one was left without a task to do in the group.

Project-based learning is a method of education that allows for a group of individuals to have the freedom in how they proceed with a given task. It allows for critical examination of said task and gives the students the freedom in how they manage their tasks.

Project-based learning is a successful method in our case because it allows each member to have the chance to further develop their knowledge of programming by trying to find solutions to the individually assigned tasks.

The method which we used during our project was the division of tasks and holding multiple meetings during the project phase and especially during the project execution. This method is most helpful because it allows for every member to have a part to play in the project, thus no one is sidelined by those of us who are more experienced with programming.

The result was approved by all members and was satisfactory to everyone's vision of the program.

6. Personal reflection:

- **Ameya Sundeep Mahankal:**

This Project was my first experience with project work with a 5-member group which felt like a daunting task was ahead of us. The team worked well together, and each member was respectful and understanding of each other and we never went beyond Friedrich Glasl's 1st level of conflict escalation. We divided work equally according to our strengths and helped each other and gave constructive feedback while meeting regularly to check the progress and to log reports in the logbook.

I thrived in the laissez-faire approach as it feels patronizing to have scheduled check-ups with regard to the project progress as we are all mature enough to make our deadlines and work at our own pace while meeting the deadline. It was refreshing to have people to rely on and reinforced my confidence in the subject by helping others and this project was a nice little introduction to a real-life work experience with colleagues. The project work also helped me keep track of all the syllabus required for the final exams apart from discrete mathematics and certain sections of RWD. Just for efficiency, I believe an approach taking stances from both Douglas McGregor's Theory X and Y could be utilized for the benefit of the students to ward off any procrastination

to build a diligent base amongst ourselves. I also learnt a few tips and tricks from my group members which actually helped with my contribution and I was really satisfied and have all agreed to go forward into the next semester with the same group. Overall, I am satisfied with the work, and my group has done and believe I have upheld myself to a great standard thanks to the SEP sessions during the semester which guided us through the documentation and work process.

- **Bozhidar Manev:**

Working in groups gives us the opportunity to experience our future day-to-day job as Software Developers. Personally in the beginning the task seemed easy but became more and more complex with the introduction of the requirements. Even though we did not have enough experience before this project, we were able to well-organise our group. One distinguishing factor from other groups is the way the work was divided between members and done within short deadlines. Every one of us had an equal position as a developer.

One of our main goals was to design a workflow and work environment, which satisfied all members. That is why we followed Maslow's hierarchy of needs. It divides all society's needs into 5 main groups. As part of our management, we succeeded in fulfilling the needs related to the project work and members’ productivity.

- **Physiological needs:** group meetings are compliant with the time of the day so members can have enough sleep. Meetings also take place in well-lighted rooms where food and drinks are allowed and taking breaks for fresh air are recommended.
- **Safety needs:** the project assures as equal as possible work division and support from others when needed.
- **Love and social belonging needs:** our group is built over the good trust and friendship between us. We are different from each other but because of our mutual acceptance, we can be time-efficient in our work.
- **Esteem**: as a group, we give each other feedback regarding the project and we support each other. Low self-esteem is the least wanted state of any of us. Even if someone has not done their task or did not understand it, the group makes sure everyone does not have any doubts about their part. Confidence, achievements and respect from others are base values of ours.
- **Self-actualization needs:** our project as part of the PBL (Problem Based Learning) path, offers a lot of problem-solving which rewards group members very well. As self-actualizing people, we have a grounded sense of well-being and satisfaction

Thanks to this hierarchy, our group was able to offer a good working environment. Everybody gave their A-game, contributed and was serious about the project. We all learned new things from each other and the work in teams. Personally, it required more soft skills and communication than I initially expected but the SEP sessions during this semester helped us with guidelines on how to manage teams, create diagrams for an optimized workflow and many more essentials.

- **Joan Tammo**:

Working on the project for me was a great chance to evaluate software engineering and the process at which software-related projects are done in general, and from my experience with the project, I have developed the sense that I am able to approach software-related projects in the future without having a sense of worry of failure because of the experiences I developed from the project.

The group agreed on dividing tasks related to projects equally among the members so no one was left without a job. This allowed the members the opportunity to grow their

skills and gain experiences, and since the group agreed on holding meetings regularly to evaluate each member's work, no one was left without feedback on their work.

The more experienced members provided and shared their knowledge with the less experienced, and the less experienced provided much-needed feedback on the work of their more experienced group members, thus creating a loop of feedback that led to the growth of every member’s skills and knowledge.

The project was planned by the group to have different phases. These phases were like goals that the group needed to achieve by working on them for a certain deadline, making motivation for the group to finish their tasks and proceeding to the next phase.

A lot of the analysis and design phases were mostly finished before the start of the project phase, and this gave us more time to work on coding and further improve the design of the program.

According to Frederick Herzberg's Two Factor Theory, there are 2 factors that affect the motivation inside a group, which are motivators and hygiene factors

Motivation factors within an education group normally are achievements, recognition, the work itself, responsibility advancement and growth.

From my experience with the group, the motivation factors were predominant because every member was strafing toward being accountable for the responsibility that they took, and thus they worked hard because achieving their part was a priority for them.

My main goal from the project was to achieve a higher degree of growth in my learning of programming, and thus I focused on fulfilling my role within the group, and this meant that I had to take responsibility for my role within the group. This meant that It was required for me to be ready to work with the tasks I was assigned, and thus fulfil my responsibilities and help achieve the group's goals.

For me, the concept of recognition within the group was intertwined with the fact that when it came to fulfilling my tasks I did not come short and fully did what was required of me, and thus I feel that I earned my comrades' recognition as someone who can be relied on.

A testament to this is the fact that I and the group have decided that we will be working together in the next semester if we were given the opportunity

Hygienic factors according to the two factors theory are group or company policies, supervision relationships with peers, work conditions, salary, status, and security.

Although some factors are obsolete in our case(such as salary, and status), from my experience the other factors did not provide any form of demotivation in the case of our group, because no member of our group had any issues with the policies that we agreed on, and our interactions with our peers were minimal, and since every member was treated equally, no hierarchy developed within the group, thus the work conditions were determined by the group as a whole and no one was given tasks that made them develop a resentment.

No amount of pressure was put upon the group because we agreed to meet up regularly and avoid delays in the project, thus we avoided any sense of insecurity that could damage the sense of progression within the group.

The factors of motivation within the group helped in eliminating any sources of conflict that may have arisen. The lack of conflict and the familiarity that the group have developed with each other helped us in eliminating any demotivation factors.

Finally, the method by which we exercised this project was favourable to me because I was not left without a part to play in the project, and I feel that I will be in a better position working with the group members in the future provided they too feel the same way.

- **Radoslav Kiryazov**

This was my first team-based project experience. I think we performed well and the team fit well together, never holding grudges and we always managed to stay level headed. We safely traversed through all of the stages of team development. And personally, I’m very proud of that. Not a lot of teams manage to get through the Storming stage of development, but everyone spoke their mind and we persevered in the end.

From the start of the project period we had the goal to not isolate the fields of work onto a singular group member. Each member had a hand in every aspect of the project, from code to documentation. The other thing I want to mention is, what I feel is, an unique approach to the waterfall method. We divided our program in different iterations. The first being with minimal functionality. And adding more functionality with every new iteration While keeping the deadlines short and the tasks small. An Important decision

our group had to make at the start was which type of team we will be according to the Douglas McGregor theory of Y and X managers. His theory doesn't fit perfectly to our current situation but can be adapted well into it and could give us valuable insight into our future jobs as Software Engineers.

I can say without hesitation that we are a Y type team. From the very start we didn't feel the urge to 'appoint a leader' if I could say it that way, as we believed in our own and the abilities of our teammates and everyone flourished when presented with responsibility.

At the end of the day I think our group did an amazing job and delivered a good product. Certainly, we had a few problems, but thanks to our SEP classes, we managed to see it through. I look forward to keeping working with my groupmates, seeing ourselves improve, with time.

- **Zsolt Nóvé**

I have had participated in team projects before but never an IT project. It was a really different experience to work either in school or from home because we were able to work at any time of the day and share the information with each other. Our group was close to each other since we started the course so we knew it could potentially be each and every member's strengths and weaknesses. But everyone had their part in the work.

The team was helpful and understanding with the others. The team never had a leader but in each scenario every member was dragging and keeping the team up and taking a leader's lore at some parts of the project inspiring us with a new idea if the old one did not work or just motivate each other generally. The group divided the project into both programming and in report writing. The group has not experienced any conflicts and we always managed to cooperate with each other.

The team made regular meetings either online or in person presenting our personal and the team’s progress. The team has made a time schedule to be sure we finish the process on time and check every point and test it until the deadline. The regular meetings were useful and progressive. In the meetings every member explained and showcased their work so far, suggesting new ideas or describing the phase of the project. Also suggesting extra features and to rework parts that we can not execute or manage to create as for now. Also the help was really helpful everytime when the member was stuck and might need a little help from the others it was always easy to find each other..

In my opinion from the Douglas McGregor theory of Y and X theory our team was strongly a Y team in the sense that all the individuals are motivated to achieve their own goals and become software engineers in the future.

7. Supervision:

The group did not depend on the supervisors, thus their effort can not be fully reflected in this report, however for the times a supervisor's input was required they did not disappoint, and thus their input is recommended in the future.

8. Conclusion:

In conclusion, this was an eye-opening experience for all members of the group. Each member displayed maturity and gave their hundred per cent to construct a program that meets all specifications within the time limit.

Overall, the group feels a great deal of knowledge has been gained through each iteration of the project and will prove invaluable as we move forward together in our education.

Some tactics to integrate in our future endeavors is to be more ambitious and to use all our time carefully. Our success may be attributed to our natural affinity amongst ourselves which helped in making the project creation a smooth process.

9. Log Book:

Date	Goal	Result
28/02/2022:	group met up to analyze the project assignment	The group agreed on a plan which is to design the classes and divide them among the group members
09/03/2022	Project description	Divided the different section of the project description
12/03/2022	Project description	Worked on the project description
13/03/2022	Project description	Finalized Project description
21/03/2022	Review the partner group’s Project description	Gave feedback on the partner group’s Project description.
23/03/2022	Project Analysis	The group made an analysis of the interview with the overlook hotel manager.
30/03/2022	Project Analysis feedback	The group worked on further assessment of the analysis of the overlook hotel interview
06/04/2022	Project Analysis 2	The group worked on the second part of the analysis phase.

13/04/2022	Project Analysis 2	The group completed working on the second part of the analysis phase.
28/04/2022	Project design	The group worked on designing the UML diagrams for the program
04/05/2022	Project design	Adjustments were made to the UML diagrams
05/05/2022	Project deign	A design for the GUI was made and agreed on by the group.
07/05/2022	Project initiation	The division of work was made, and each member was to work on a section of the program
10/05/2022	Project phase	The group worked on the project
14/05/2022	Project phase	The group finished the first iteration of the program, and worked on the GUI
16/05/2022	Project phase	A testing of the program's first iteration was made, and a merging of the different classes was made and a testing of the program was done, and work began on the second iteration of the program, and further development of the GUI was done taking in consideration the input of the group.

19/05/2022	Project phase	A search method for the rooms was discussed and with the input of all members in consideration the work on a search method was initiate And the work on the second iteration of the program began taking in consideration the development of a discount method, and further development of the reservation class
22/05/2022	Project phase	A search method was developed and was implemented, and work on a search function for the website and a design for the website began.
25/05/2022	Project phase	Finished the website and the GUI and the coding of the program and started on documentations.
28/05/2022	Project phase	Worked on the project report And the user guideline
30/05/2022	Project phase	Worked on the project report And finished writing the user guideline
31/05/2022	Project phase	Worked on the project report and started working on the process report

01/06/2022	Project finalization	Finished process report and project report.

Project report: The Overlook Hotel

Ameya Mahakal (326157)

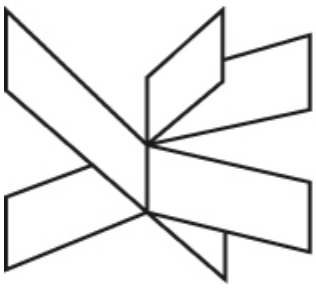
Bozhidar Manev (326391)

Joan Tammo(325753)

Radoslav Kiryazov(326155)

Zsolt Nóvé (326345)

Supervisor: [Allan Henriksen , Mona Wendel Andersen]



VIA University College

[Number of characters: 17479 (without spaces)]

[Software Technology Engineering]

[Semester 1]

[01/06/2022]

Table Of Contents

Abstract

The goal of the project is to create a booking system for The Overlook Hotel utilizing the waterfall method.

The program was analyzed and designed with the use of UML diagrams resulting in the development of a plan for the development of the program.

The project is created using Java, JavaFX HTML5, CSS and Javascript. It consists of a Java booking program, and a website that provides information about the Hotel. The website for the hotel provides a searching functionality which uses XML files and JavaScript functions to operate.

Although many of the specific requirements were not implemented, the majority of the general requirements were.

The program is capable of exercising many of its general functionalities, however many of the functionalities were not implemented in the GUI due to the lack of experience of the group member with GUI implementation.

1 Introduction

The main purpose of this program is to develop an offline single user system for a hotel in order for its staff to better manage guest data and reservation statuses.

The system developed is a simple program with a detailed manual to help the staff get in terms with its functions and limitations.

The customer is The Overlook Hotel management who previously worked with a simple ledger which often led to confusion and inconveniences among guests.

The project also contains the room status of all **booked** rooms of the hotel on the official website.

The project is made by a group of five members utilizing Java, FXML, HTML, CSS, as well as JavaScript.

The requirements of the program were discerned from a transcript between Mr Stuart Ulman, the manager and a representative.

The report contains the many phases established during the process of the project such as analysis, design, implementation,tests results and conclusions in detail that follow below.

2 Analysis

The analysis phase was based on an interview with the manager of the Overlook Hotel, in which they required the construction of a program that helps with the booking of reservations at the hotel.

From the interview, the following analysis was made:

2.1 Requirements:

Functional requirements:

Critical Priority:

1. As an employee, I want to make a reservation for a given interval, guest, start date, and end date.

2. As an employee, I want to search for rooms by room type, and smoking availability.
3. As an employee, I want to register a guest with name, address, phone, nationality, check-in and check-out.
4. As an employee, I want to add, modify, display or delete reservations.
5. As an employee, I want to add, display or delete data about guests and rooms.

High Priority:

1. As an employee, I want to modify the price of rooms.

2. As an employee, I want to display the total amount to be paid by a guest during their stay.

3. As an employee, I want to check room availability on a specific date.
4. As an employee, I want to add an additional bed, if the guest has requested it.

Low Priority:

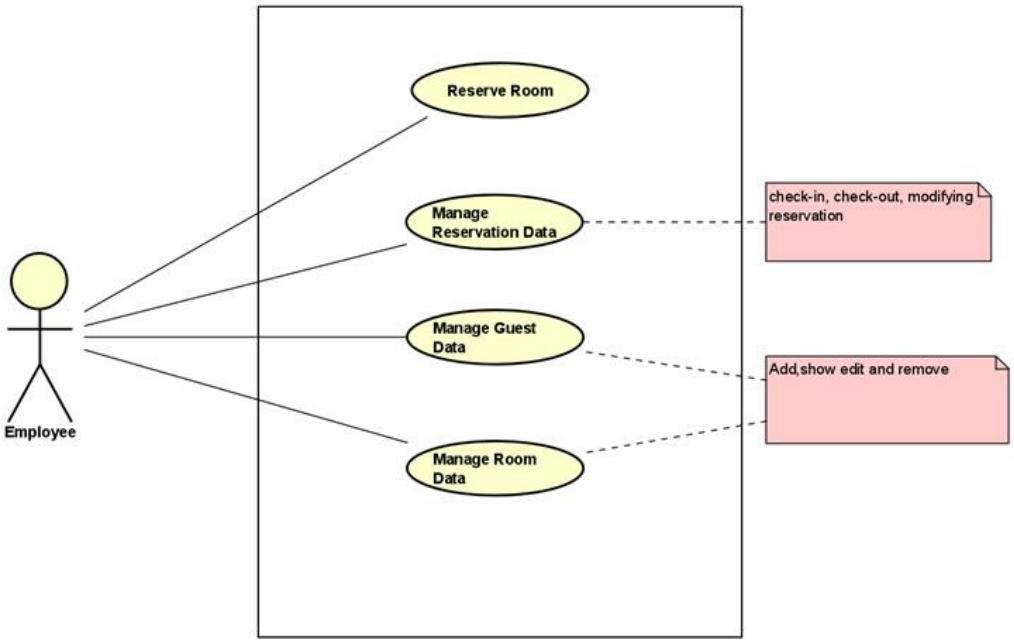
1. As an employee, I want to add, modify, display or delete data about rooms,so rooms can be created, the information about the rooms can be modified or deleted and room information can be displayed.
2. As an employee, I want to update the rent price for the rooms, so that the prices of the rooms can be corrected.
3. As an employee, I want to set a room to unavailable if the room is undergoing refurbishment.
4. As an employee, I want to specify if smoking is permitted in the room, so the guest will be able to smoke in the room.
5. As an employee, I want to add an extra smoking fee for customers.
6. As an employee, I want to search for rooms that are available for smoking.
7. As an employee, I want to search for guests that are allowed to smoke.
8. As an employee, I want to search for frequent guests.

Non-functional requirements:

1. The program will be made in Java language.
2. The program can interact and input can be added by mouse or keyboard
3. The program will be easy and simple to use for the employees
4. The program is available for the employees when the reception has opening hours.

2.2 Use case diagram:

The use case indicates what the user wants the program to do. The following diagram shows the required functions of the program:



The diagram shows the interactive options between the user and the program. The user will be able to reserve rooms and manage reservation data by adding or deleting reservations. The management of the guest data is about registering the guest's information in the program and storing the information for future use for discounts.

The management of the room data is about indicating how many rooms will be reserved for a given period of time.

2.3 Use case description:

The guest management is the use case description that was chosen for this analysis.

Use case	**Manage Guests**
Summary	**Add, modify or delete data regarding guests**
Actor	**Employee**
precondition	Guest data is not up to date.
postcondition	**A customer has been added and is being assigned a room or removed from the list of customers** Guest information has been updated, and guest has been removed or added, to said booking.

Base sequence	**ADD:** **Adding new customer with values:** **1a)Name(String)** **1b)Phone(String)** **1c)Address(String)** **1d)Check-in-Date(String)** **1e)Check-out-Date(String)** **System validates data entered and if wrong asks user to re-input data** **DISPLAY:** **2a)Show a list of all customers** **2b)Show a list of customers between two dates** **EDIT:** **Edit values of customers information** **DELETE:** **Delete a customer after verifying action** 1. Search Booking 2. Check guest's name, phone number, and address 3. Booker provides new Information. 4. Edit the guest with the new details provided by the guest. 5. Update the guest information and save.

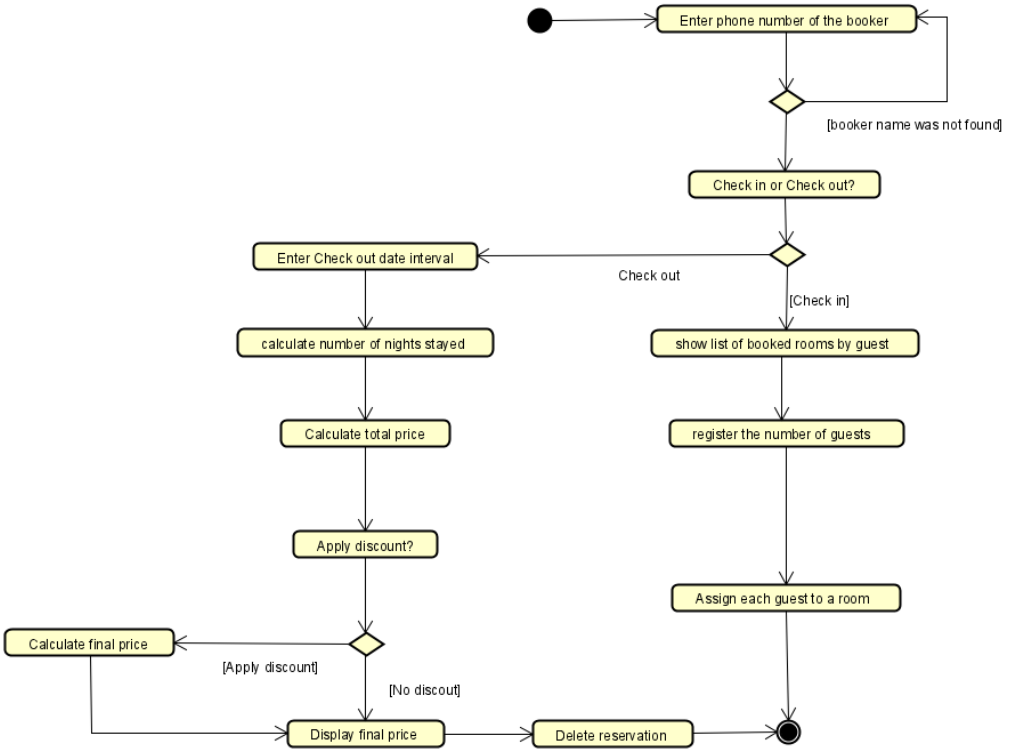
Exception sequence	REMOVE GUEST Run Base Case(1-2) 3. Remove guest(s), provided by the booker from the reservation. 4. Update the guest in the booking and save. ADD GUEST Run Base Case(1-3) 4. Create a new guest 5. Add the guest to the booking and save.
note	

In the use case description the base sequence shows the process at which the registration of a guest will be done. The main functionality of guest management is to add, remove or edit the information of a guest. The use case description illustrates the steps and the sequences at which these processes are done.

2.4 Activity diagram:

The activity diagram chosen to be in this report is the reservation activity diagram

The reservation process is properly illustrated in the diagram:



The reservation process takes into consideration whether the guest would be checking in or checking out.

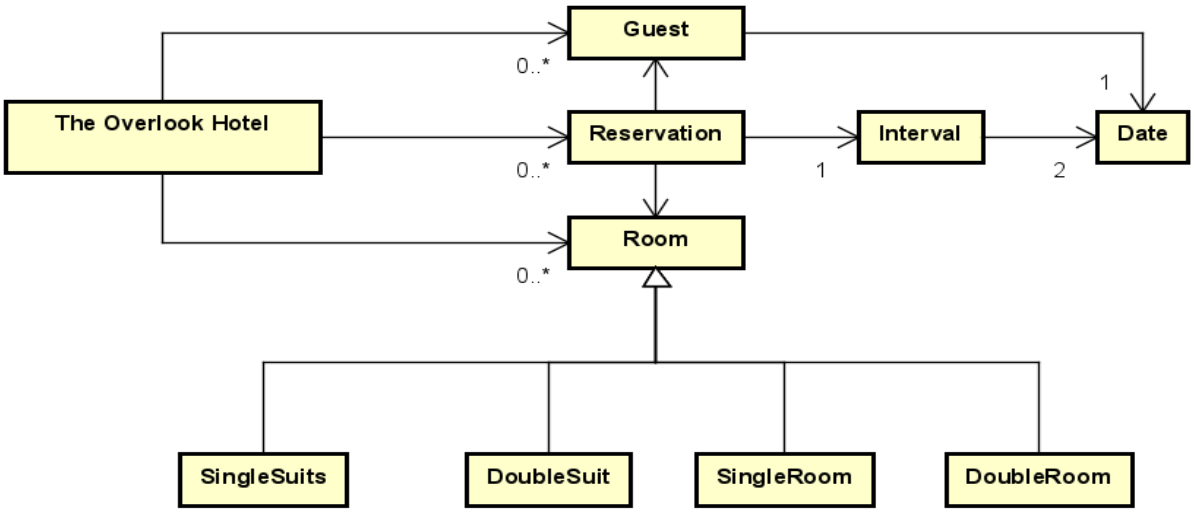
In case of a check the guest will be asked to indicate the check-in and check-out date, thus a reservation would be created and registered in with a phone number.

In case of a check out the user will indicate the date of the checkout, and whether a discount should be applied or not, if so, the user would be asked to enter the discount, and in either case, the final price will be indicated and the reservation would be deleted.

2.5 Domain model:

The domain model illustrates the relationship between the main classes of the program. The model provides an idea of how the different parts of the program are connected, and it illustrates the dependency relationships between the classes.

The following is a domain model for the Overlook hotel reservation program:



The model shows that the reservation will require a time interval indication which will be the check-in and the checkout. The interval class will require the registration of 2 dates: the check-in and check-out dates.

The room class will contain 4 types of rooms, and users can assign a check-in and check-out date to one or multiple rooms.

One or multiple rooms may be reserved at a time by one guest.

One or multiple guests may reserve the same room but at different and non-conflicting time intervals.

The guest will be assigned one birthday from the date class.

One guest can make one or multiple reservations, and one or multiple reservations may be made by multiple guests.

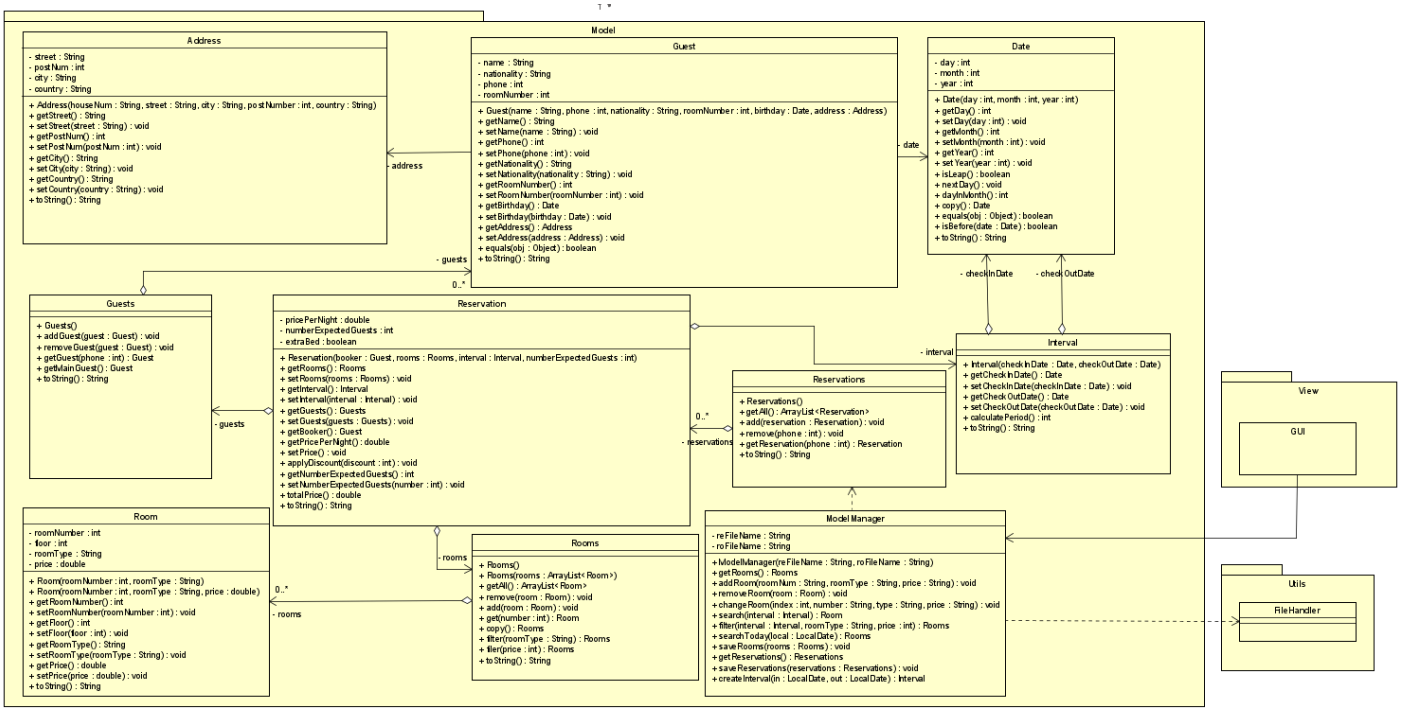
3 Design

The design phase focuses on creating an approach for the coding phase.

The purpose of the design is to develop a plan that the group can come back to during the implementation phase.

3.1 Class Diagram:

The class diagram illustrates the different classes and their relations with each other:



The class diagram is divided into 3 packages:

1. A **utility package** containing the FileHandler class responsible for the management of the files.
2. A **view package** that contains the GUI (Graphical User Interface) class that works as a terminal and helps the user navigate the functionalities of the program.
3. A **model package** that contains the different classes that are responsible for the different functions of the program.

The model package contains classes responsible for the program's main four functionalities, which are:

1. The management of time intervals
2. The management of rooms
3. The management of the guests
4. The management of reservations

The class diagram illustrates the dependency of these 4 functionalities on one another, for example, a guest requires a birthday from the date class, the reservation requires instances of the Interval class, the Guests class, and the Rooms class.

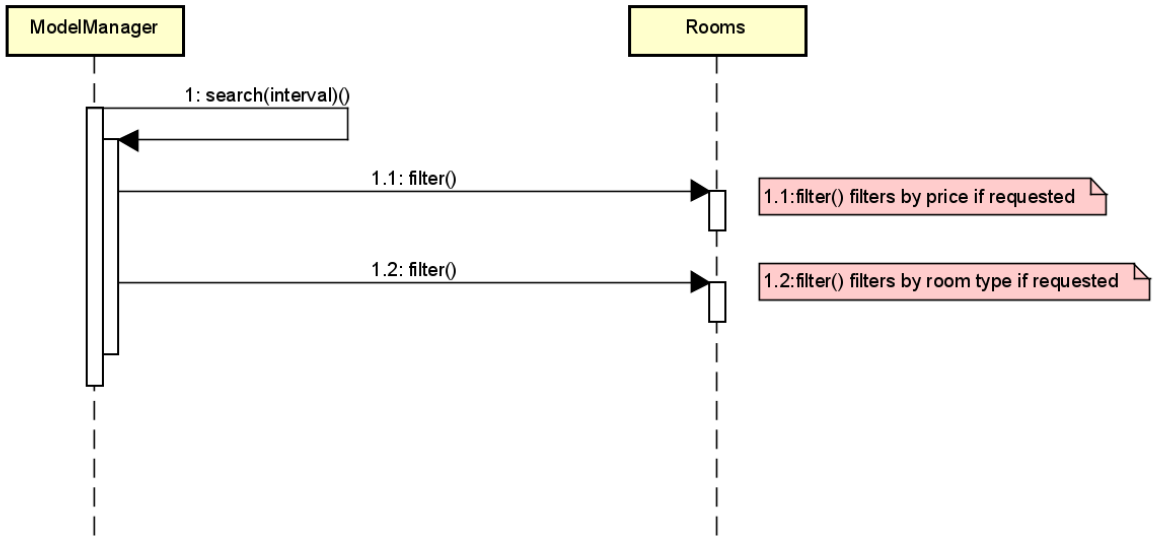
The file handler provides a connection with external files.

The GUI is dependent on the model manager, as the only access point to the model package, to access the functionalities of the program and import data from the external files.

The relations between classes are the result of the group's well-thought organization which follows the good practice of low coupling and high cohesion. This organization makes the code maintainable.

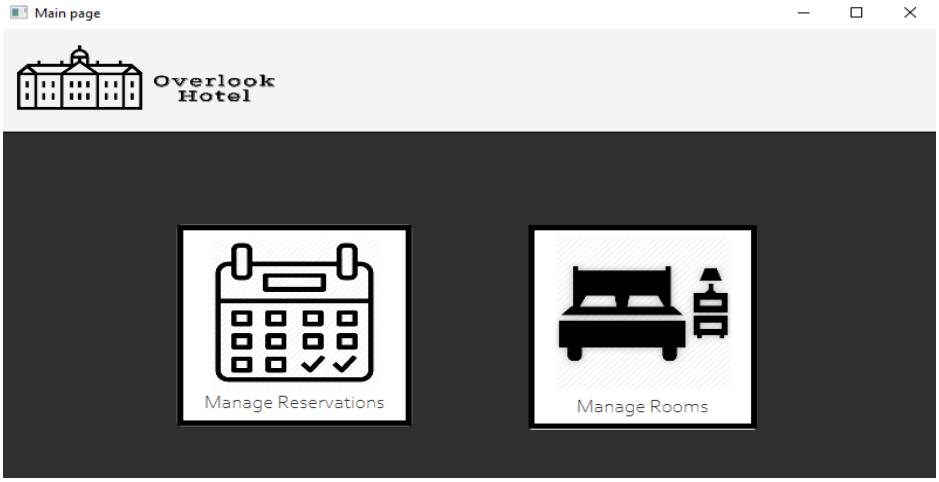
3.2 Sequence Diagram:

When calling the method, a method search() is called which returns a Rooms object with all the rooms available for the given pair of check-in and check-out dates. If a price filter is entered, a method filter(price) is called, which filters the available rooms returned from the previous method. It is possible to apply a third filter for room type. In this case, a method filter(room type) is called which further filters the available rooms returned from the previous methods.

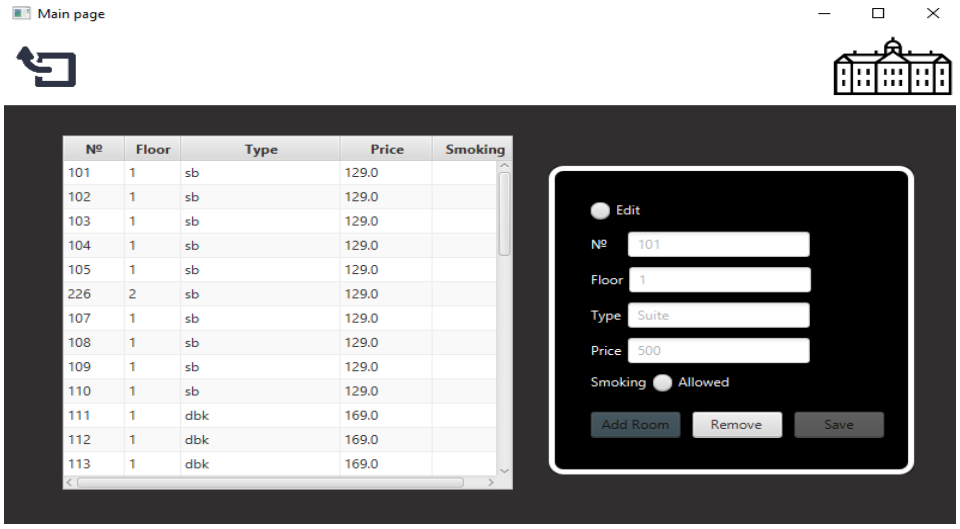


3.3 GUI:

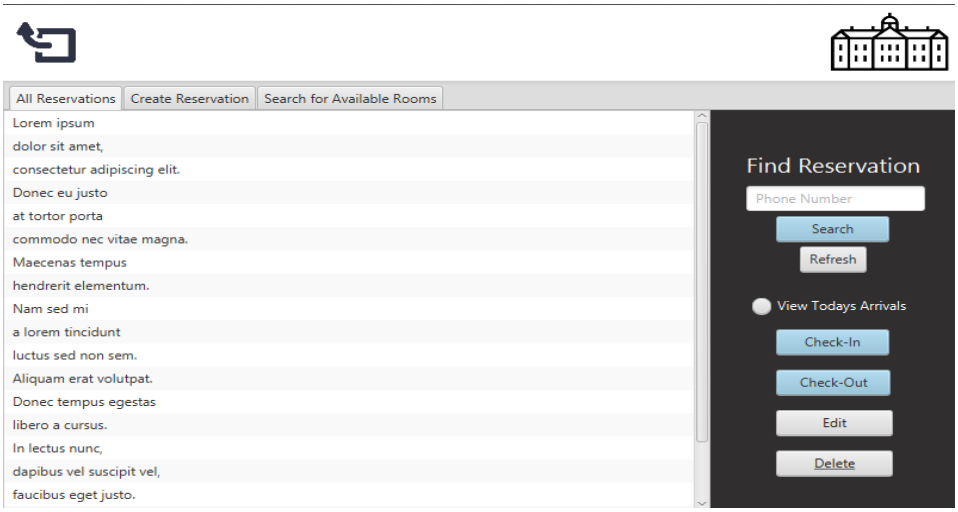
Upon launching the program you are presented with a simple two-button UI. The Controller class of this scene has two methods switchToManagingRooms() and *switchToReservations()* each assigned to their corresponding button.



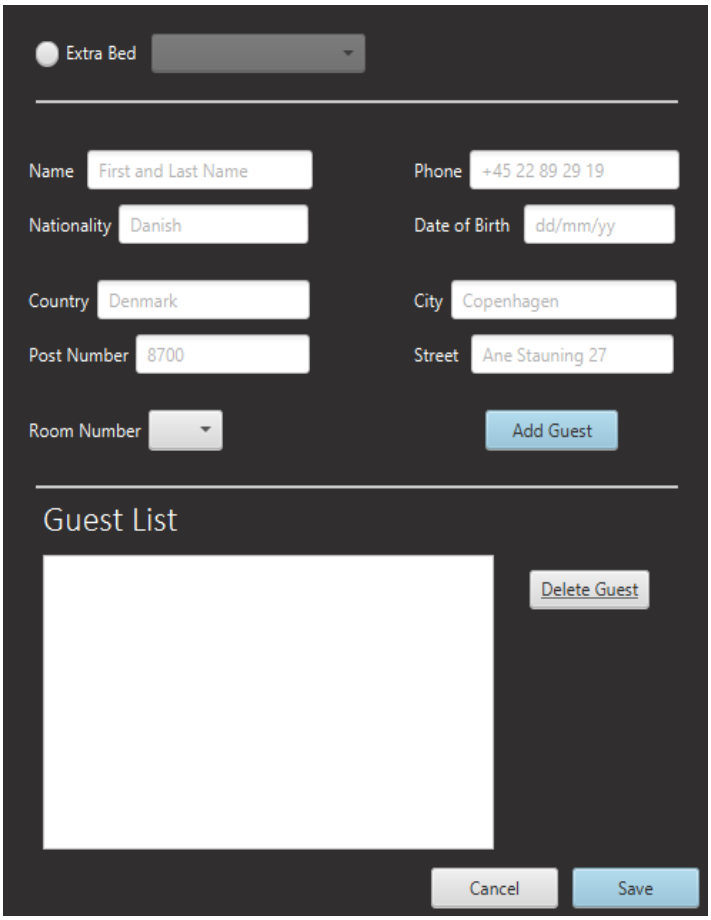
In the Manage Rooms tab, an Observable List which contains all of the rooms is presented on a table-view. The text fields remain non-editable until the radio button is selected. The user can add a brand new room to the list, or select an already existing one from the list and edit or delete it entirely.



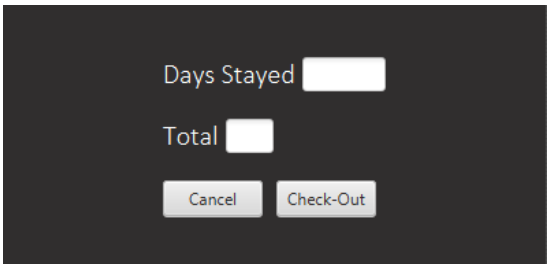
Upon entering the reservations page, update methods are called in the background, refreshing all of the lists. The search function filters through the reservations looking for a booker's phone which matches with the user input and updates the list. View Todays Arrivals looks through the reservations and updates the list with the ones which have check-in dates matching the current date.



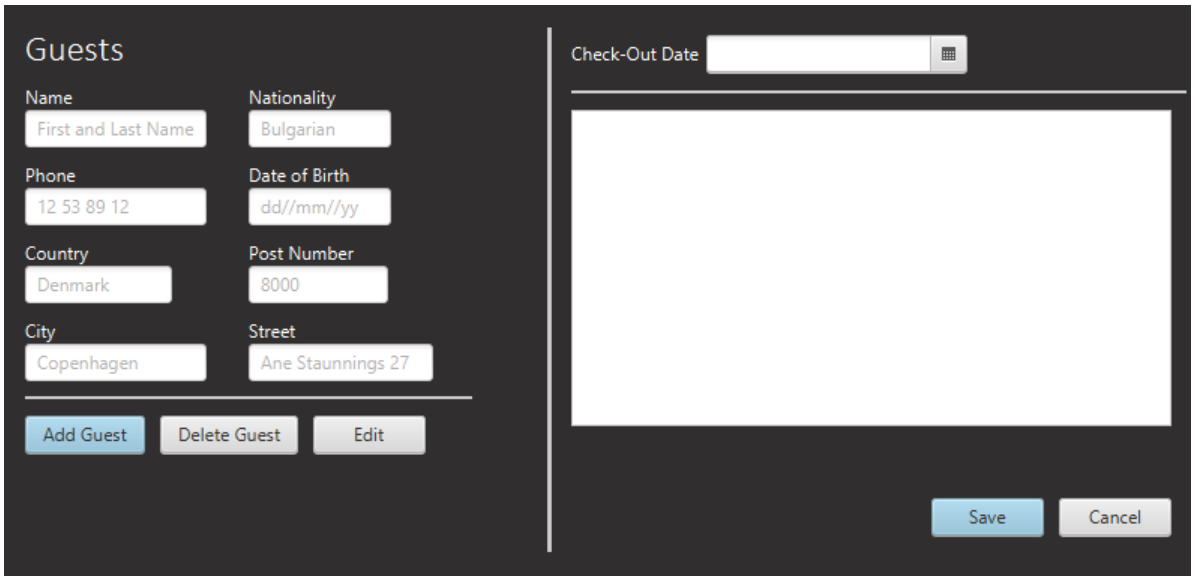
When checking in a guest a window shows up prompting the user to input the guest information, and assign a room number and an extra bed *(optional).*



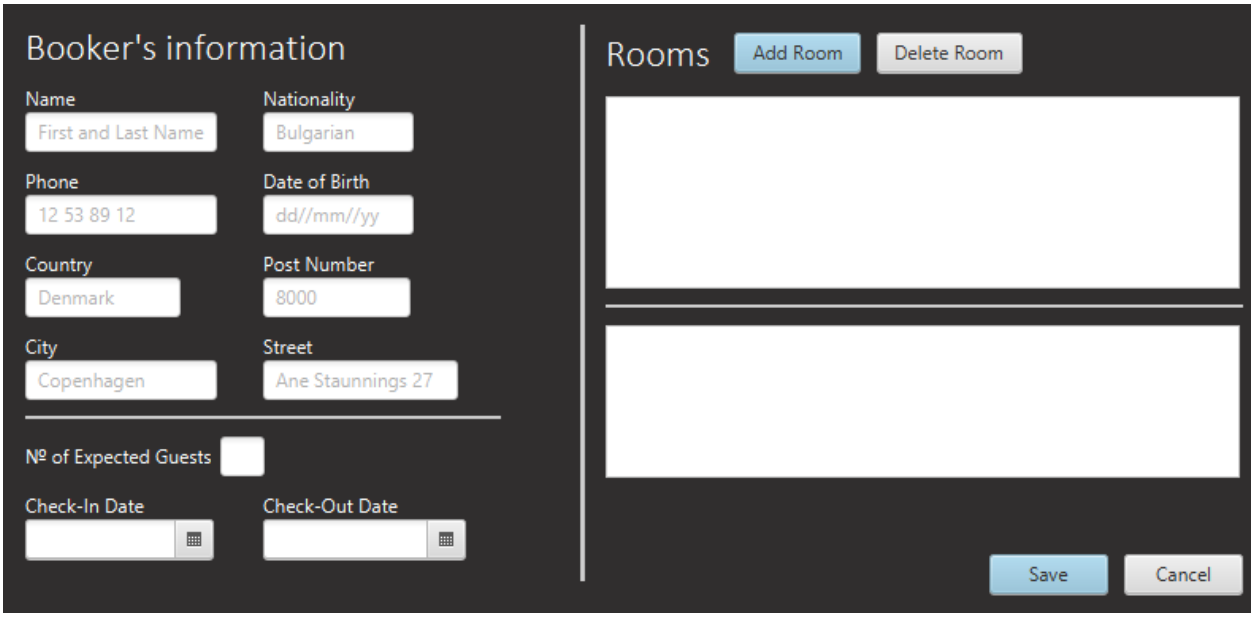
Check-out shows the time stayed and calculates all the guests' expenses and deletes the reservation after confirmation.



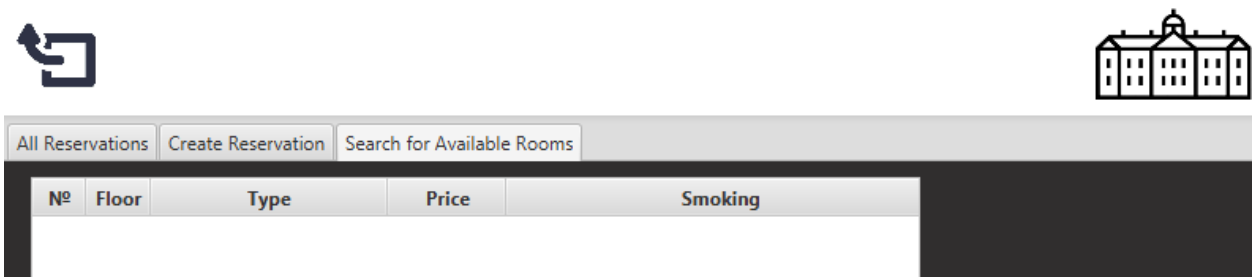
When editing the reservation, you can change all the fields of the guest and the check-out date



Creating a reservation requires only the information of the booker, check-in and check-out date. The entire list of rooms is displayed on top, from which you can add onto the small room list in the reservation. From here the reservation is stored on a binary file and the whole reservation table is refreshed



Searching for Available rooms filters through the rooms , with an given interval, looking for a free room. Create Reservation sends you back to the tab with the Rooms already selected.



3.4 Website:

Website execution :

The goal of the website was to create informative pages that show the basics and uniqueness of the hotel. Clearly tells the price and the rules with colorful pictures and furthermore, the history and the information of the staff and their email for the customer in order to gain further information if necessary.

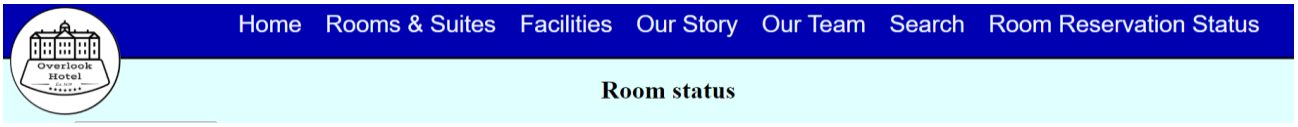
For the project we used w3school as a help and many other sites to gain information about the structuring of websites.

The group has divided the work between the five members and therefore the site contains five html sites for the site basic look and one extra html file connected with a javascript using a search function. All the pages are using different css file as well with some basics matching just as the top menu, scroll bar and social media availability.

On every site of our webpage the team has created a top and a bottom menu and a logo that has been used for each and every site that we have made.

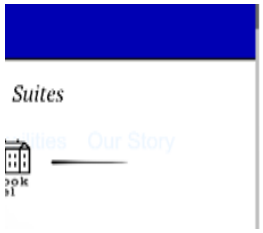
On every site of our webpage the team has created a top and a bottom menu and a logo that has been used for each and every site that we have made.

Top menu bar :



Bottom menu bar :

The scrollbar



Also a new scroll bar has been added to all the sites to look different from the generic browser scroll bar.

These two were made with a list. The navigation bar contains it and in the lists, the other html sites are being held. In the footer the social media links have also been added in a division. Both the header and the footer contain a logo image which is stored in a division.

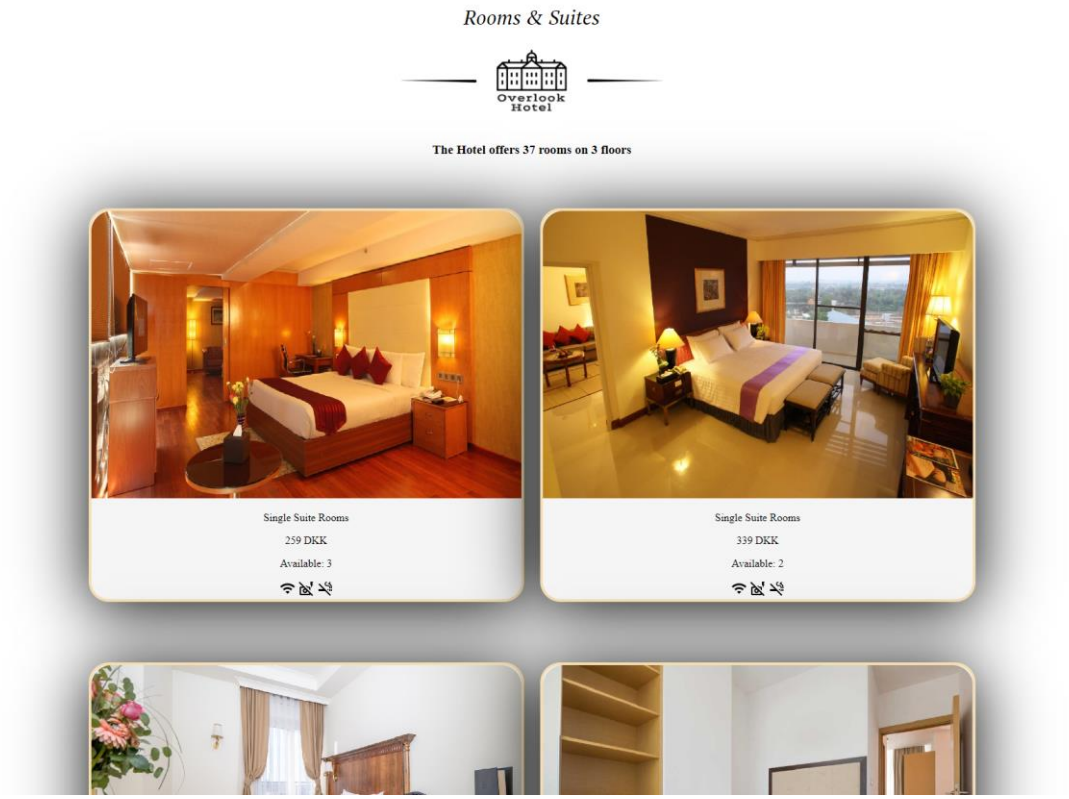
Front page

At the beginning in order to make the site the team divided the group work amongst each other and we created five web pages at first in total.

The sites consisted of a main page where we have created a little description of the hotel and also provided some gallery pictures to it. And the list of the rooms is mostly made of divisions with pictures stored in them with some description about the hotel.

The second site in the row is the Rooms and Suites where we have placed the selection of the hotel rooms with pictures from the interior and some extra information such as the pricing, availability and basic rules for the room.

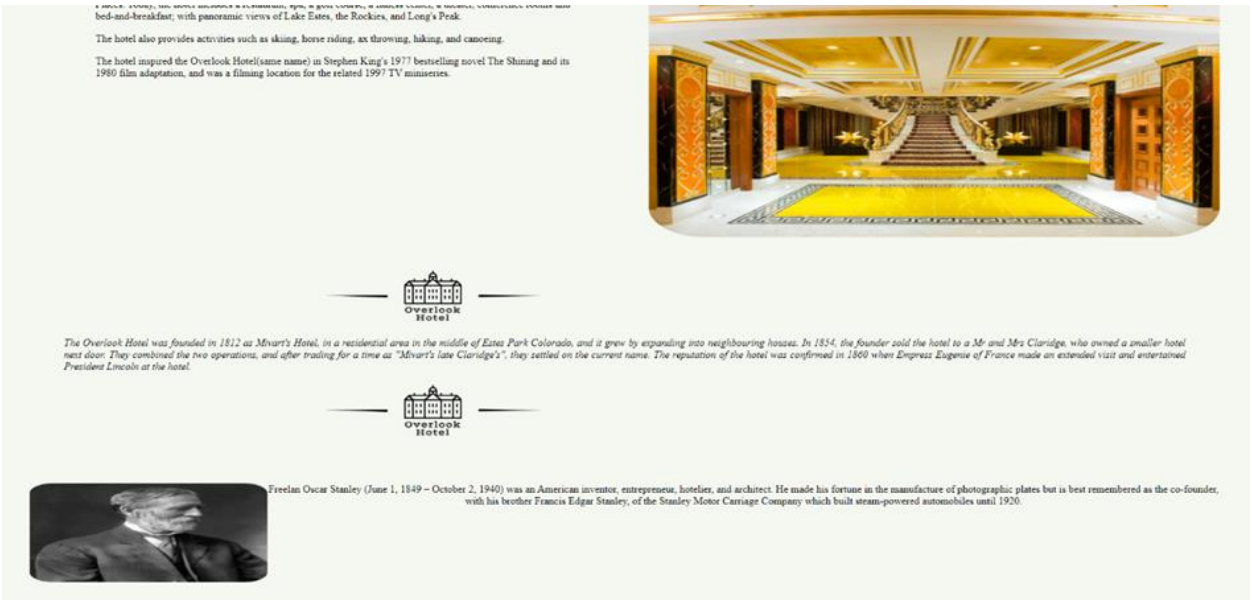
Another addition is that if you go to the rooms section and hover over your mouse on the pictures they will pop out and have a slightly different color.



And on the last page we have created our team where all the contact information can be found about the hotel and the current staff of the hotel.

On the facilities page, we have placed some photos and information about the unique extras that the hotel offers such as the golf course, the restaurant and bar, the theater, conference rooms and others. In the our story section we have created a little background description of the hotel supported by some in and outside photo of the hotel furthermore, some information about the founder of the hotel with a portrait with it.

Our story and facilities are build from divisions and paragraphs. On the facilities page, we have placed some photos and information about the unique extras that the hotel offers such as the golf course, the restaurant and bar, the theater, conference rooms and others. In our story section we have created a little background description of the hotel supported by some in and outside photos of the hotel furthermore, some information about the founder of the hotel with a portrait with it.



The addition of two search functions has been made on a separate site for the website. On the first site, you can

search for rooms by either room type, room number or price.

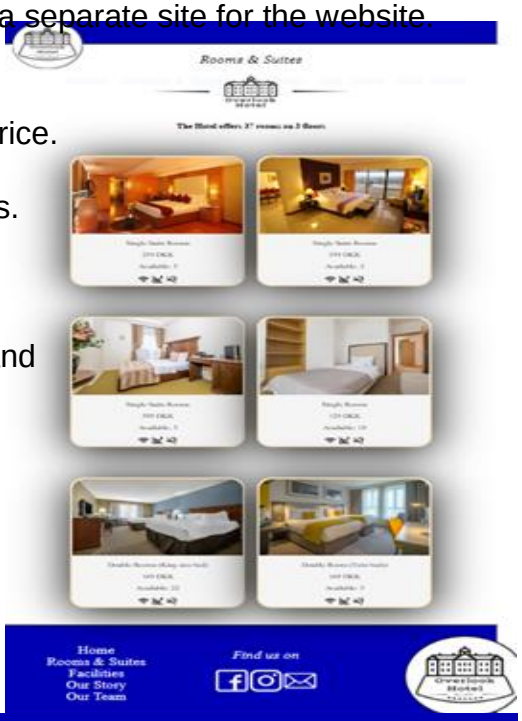
On the second you can check already made reservations.

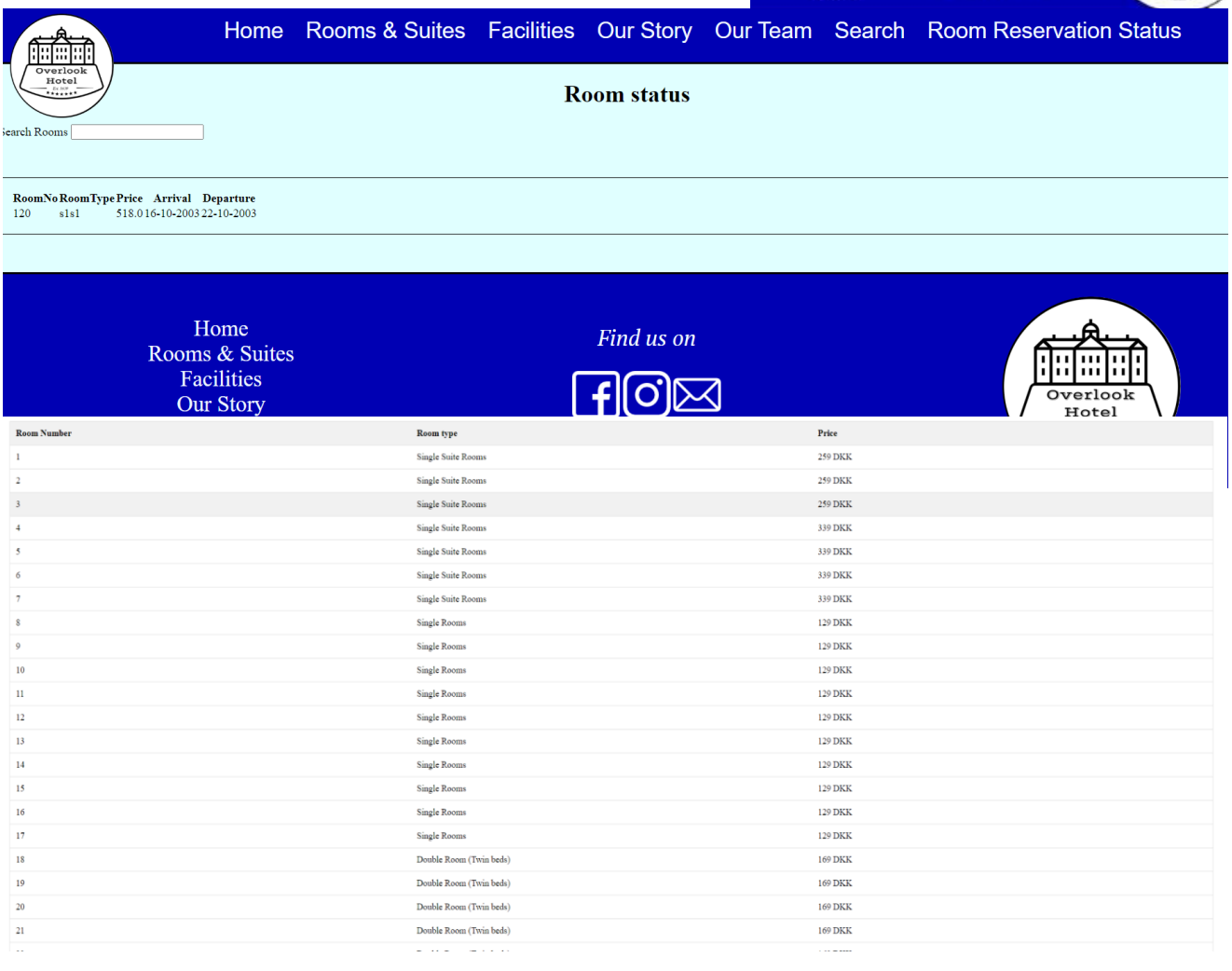
On this site, we have used javascript in order to search

separately from the XML file for the made reservations and

search for rooms. In order to search just type

numbers or letters in the search bar.





Room Number	Room type	Price
1	Single Suite Rooms	259 DKK
2	Single Suite Rooms	259 DKK
3	Single Suite Rooms	259 DKK
4	Single Suite Rooms	339 DKK
5	Single Suite Rooms	339 DKK
6	Single Suite Rooms	339 DKK
7	Single Suite Rooms	339 DKK
8	Single Rooms	129 DKK
9	Single Rooms	129 DKK
10	Single Rooms	129 DKK
11	Single Rooms	129 DKK
12	Single Rooms	129 DKK
13	Single Rooms	129 DKK
14	Single Rooms	129 DKK
15	Single Rooms	129 DKK
16	Single Rooms	129 DKK
17	Single Rooms	129 DKK
18	Double Room (Twin beds)	169 DKK
19	Double Room (Twin beds)	169 DKK
20	Double Room (Twin beds)	169 DKK
21	Double Room (Twin beds)	169 DKK

4 Implementation

The method chosen for the complexity analysis was the search method because it covers the critical priority requirement of searching for room types. The search method includes 5 other methods:

1. `iseBefor` compares two date intervals and declares if one of the dates is set prior to the other one.

```
public boolean isBefore( @NotNull Date date) {
    if (this.year >= date.year) {   // 4 time complexities
        // if the years are not equal
        if (this.year > date.year) { // 3 time complexities
            return false; // 1 time complexity
        }
        if (this.month >= date.month) {     // 4 time complexity
            // if the months are not equal
            if (this.month > date.month) {  // 3 time complexities
                return false;   // 1 time complexity
            }
            // return the boolean result of day comparison
            return this.day >= date.day; // 5 time complexities
        }
    }
    return true;    // 1 time complexity
    // O() = 22 = O(1)
}
```

2. The `filter(int price)` method filters the rooms by the price and returns a `filtered list.`

```
public Rooms filter(int price) {
    // Creating a copy of the Rooms object with which we will work.
    Rooms filteredList = this.copy();   // 2 time complexities
    // Removing all rooms which do not comply with the given room type.
    filteredList.getAll().removeIf( filter: room -> room.getPrice() > price);    // 3n + 1 time complexities
    return filteredList;    // 1 time complexity
    //O() = 3n + 4 = O(n)
}
```

3. The `filter(int price)` method filters the rooms by the price and returns a `filtered list.`

```java
public Rooms filter(String roomType) {
    // Creating a copy of the Rooms object with which we will work.
    Rooms filteredList = this.copy();  // 2 time complexities
    // Removing all rooms which do not comply with the given room type.
    filteredList.getAll().removeIf( filter: room -> !room.getRoomType().equals(roomType)); // 3n + 1 time complexities
    return filteredList;    // 1 time complexity
    //O() = 3n + 4 = O(n)
}
```

4. The `Search` method uses the information from the two files(`reservation file` and `room file`) to show the available rooms for the given time interval(`check-in and check-out dates`)

```java
public Rooms search(Interval interval) {
    // a copy of the Rooms object will be used for editing
    Rooms available = getRooms().copy();    // 2+n time complexity
    // Iterating through all the reservations from the file
    for (Reservation reservation : getReservations().getAll()) {    // n^2 time complexity
        /*if the given check-in and/or check-out date clashes with check-in
        and/or check-out dates of already created reservations*/
        if ((!interval.getCheckOutDate().isBefore(reservation.getInterval().getCheckInDate()))
                || (!reservation.getInterval().getCheckOutDate().isBefore(interval.getCheckInDate()))) {
            // 8 time complexities
            // Remove all rooms reserved for the given reservation from the Rooms object we are using
            available.getAll().removeAll( c: reservation.getRooms().getAll()); // n^2 time complexities
        }
    }
    return available; // 1 time complexity
    // O() = 2+n + n^2 + 8 + n^2 = O(n^2)
}
```

5. The `filter` method in the `Model Manager` class shows available rooms, filtered by specific filters(as `time interval, price` and/or `room type`). The method can work with one as well as with the three filters.

```java
public Rooms filter(Interval interval, String roomType, int price) {
    //  First return all available rooms for a given Interval and filter if requested
    Rooms available = this.search(interval);    // 1 + n^2 time complexity
    // if a preferred price was given then filter by price
    if (price != 0) {   // 1 time complexity
        // filtering by price
        available.filter(price); // O(n)
        // if a preferred room type was given then filter by room type
        if (roomType != null) {
            // filtering by room type
            available.filter(roomType); // O(n)
        }
    }
    //returning all available rooms after filtering
    return available;
    // O() = 1+n^2 + 1 + n + n = O(n^2)
}
```

5 Test

The purpose of the test section is to document the result of your testing; to verify if the content of the requirements section has been fulfilled. How is the system tested, and which strategy has been used; e.g. White Box (Unit Test), Black Box, etc.

Use Case	Testing Result	Comment
Room Reservation	Success	The program can make reservations and save them into a binary file and read from it
Enter room type, check-in and check-out date	Success	The user can add check-in and out date. The user can add one or more rooms to the reservation
System shows all available rooms for the period	Failed	The code is available and tested(working) but it isn't implemented into the GUI
Select room from the list	Success	The program is able to add one or more rooms to the reservation
Add extra bed if guest demands	Failed	Implementation inside GUI missing
Adding guests to the reservation	Failed	Implementation inside GUI missing
If the guest does not exist in the system database: create a new guest	Missing	The group didn't implement the code due to the design changes after submitting of the requirements
The reservation process can be cancelled at any time	Failed	A cancel method is not available
Manage Reservation Data	Failed	Check-in and out code is present but not implemented into the GUI

Enter the full name, phone number or email of the guest	Success	Success only for the booker(first guest)
The program shows the status of the booking for the guest	Missing	The group didn't implement the code due to the design changes after submitting of the requirements
The request of the guest such as new bed sheets, or a request for deliverables such as food or beverages will be registered in the system.	Missing	The group didn't implement the code due to the design changes after submitting of the requirements
The request for the change of rooms will be done by the receptionist.	Failed	Rooms can be changed for the reservation. Code for changing rooms per guest is available, but not implemented into the GUI
Manage Guests	Failed	The code is present but not implemented into the GUI
Adding new guests with values: -Name -Phone -Address -Check-in-Date -Check-out-Date	Failed	The code is present but not implemented into the GUI
System validates data entered and if wrong asks the user to re-input data	Failed	The group did not manage to add validation statements into the input methods
Show a list of customers between two dates	Missing	The group didn't implement the code due to the design changes after submitting of the requirements
Save a customer after	Missing	The group didn't implement the

verifying action: 1. Search Booking 2. Check guest's name, phone number, and address 3. Booker provides new Information. 4. Edit the guest with the new details provided by the guest. 5. Update the guest information and save.		code due to the design changes after submitting the requirements
Manage Room Data	Success	The program can manage rooms(edit, add, delete). All the code is working properly and works well with information from a binary file

In conclusion, the testing phase was the most or second most intense phase of the whole project. The group was able to code the majority of the methods, but unfortunately, the group faced obstacles during the merging of the Java code and the GUI one. The lack of experience with GUI led to the shortcomings of the final result.

6 Results and Discussion

The project was able to achieve some of the required general functionalities, such as the reservation of rooms, and searching for a given reservation with a phone number.
The program can work properly with files(such as reading and saving of rooms' and reservations' information).
The shortcomings of the program include most of the specific requirements(review appendix 2131)

Bad requirements, taught by the group, in the beginning when the group members were not experienced enough, are the reason for the majority of the tests being "Missing". At the end of the project, the group members are more experienced and realize how the majority of requirements are meaningless.

7 Conclusions

The group went through the analysis phase and design phase with no obstacles, however when it came to the implementation phase and due to the lack of experience with the GUI the group was unable to implement many of the coded methods inside the GUI, thus resulting in the lack of many specific requirements that were established during the analysis phase.

8 Sources of information

Note: Use the standard reference method: Harvard Anglia. A very good reference tool is Mendeley (Mendeley.com 2016), **ask VIA Library if you need help.**

Jon Duckett. 2011. HTML & CSS Design and Build Websites. Tony Gaddis. 2015. Starting out with Java-Early Objects. 5th ed. Haywood Community College

Kenneth H. 2012. Rosen. Discrete Mathematics and its Applications. 7th ed. Monmouth University

Jon Duckett. 2011. HTML & CSS Design and Build Websites.

10 Appendices

Appendix-1-ClassDiagram

Appendix-2-Activity-Diagram_ManageGuests

Appendix-3-Activity-Diagram_Reservation

Appendix-4-Activity-Diagram_Reservation-CheckIn-And-CheckOut

Appendix-5-Domain-Model

Appendix-6-Sequence-Diagram

Appendix-7-log-Book

Appendix-8-Project-Description

Appendix-9-Functional-Requirements

Appendix A Project Description:

[Project description: Overlook Hotel]

Ameya Mahankal, 326157

Bozhidar Manev, 326391

Joan Tamm, 325753

Radoslav Kiryazov, 326155

Zsolt Nóvé, 326345

1. Background Description

Overlook hotel is a small family-owned business that has unfortunately fallen behind the times. Their booking management method is outdated, causing the staff and the guests unnecessary confusion and problems, which could be easily avoided. The aim of the project is to modernise the already existing system, used by the hotel(client), with a new, more reliable and easy to use single user offline system managed by one person on a single computer.

The hotel currently uses a ledger staffed by the receptionist at the front desk to book reservations for its guests. This is proving to be a challenge for the staff to manage and organise, which could lead to inconveniences for the guests. It could also impact the future reputation of the hotel.

Although there are cloud-based programs that are meant for booking, they are prone to data loss, making them less reliable and appealing alternatives for their current methodology.

The hotel has 32 rooms and 5 suites over two floors of various sizes and prices which can also accommodate additional requests such as extra bed, smoking, late arrival etc[1].

2. Problem Statement

The hotel is losing business due to the lack of a proper booking system. The limitations of the ledger are already reached.

[1] Transcript of Overlook Hotel interview

Issues:

- Registering the details of the guest(name, home address, date of birth, nationality)
- Managing guests' details
- Displaying a list of free rooms
- Assigning rooms to each guest
- Adjusting the price of each room
- Managing the demands of the guest(smoking, discount, changing rooms, late arrival, extra bed etc)
- Avoiding double-booking of rooms
- Handling check-in and check-out of guests
- Displaying a list of guests that will arrive on a certain day and with assigned room

3. Definition of purpose

The purpose of the program is to facilitate the booking process, reducing the number of errors caused by the lack of a precise method for room booking in Overlook Hotel.

4. Delimitation

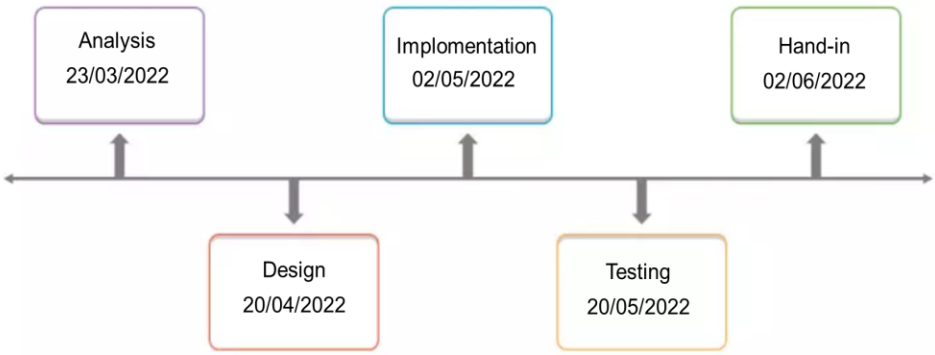
1. Not developing a multi-user system.
2. Not keeping a historical archive of bookings and guests.
3. Not developing an online booking system.
4. Not developing a cloud-based program.

5. Methodology

An analysis of the problem will be done to help in the designing of the UML diagrams.
By using the diagrams as a guide, the process of implementing the functions of the program will begin.
Multiple testing sessions will be done to detect problems in the implementation process of the program.
After the analysis, the process of designing the UML diagrams in Astah will begin. The UML diagrams will showcase the proper number of classes, their instance fields, and the corresponding methods within them.
The implementation process will be done in IntelliJ. The program in IntelliJ will be constructed based on the UML diagrams. The implementation process will be divided by the number of designed classes between the members of the group. Each member will take the responsibility of building a part of the program.
Multiple meetings will be held for testing the program and fixing the problems related to the implementation process.

6. Time schedule

(Using the waterfall model)



Analysis

23/March/2022

Approximate hours spent : 125h

Design

(20/April/2022)

Approximate hours spent : 325h

Implementation

(02/May/2022)

Approximate hours spent . 450h

Testing

(20/May/2022)

Approximate hours spent : 230h

Handing-in

(02/June/2022)

Approximate hours spent : 20h

7. Risk assessment

Risks	Likelihood Scale: 1-5 5 = high-risk	Severity Scale: 1-5 5 = high-risk	Product of likelihood and severity	Risk mitigation e.g. Preventive- & Responsive actions	Identifiers	Responsible
Scope Creep	3	4	12	We will regularly review the scope of the project to ensure we do not develop unnecessary functions	Lack of project management practices	Bojidar Manev
Difference between the design of the program and its implementation in IntelliJ	4	4	16	Regular comparisons between code and its implementation in IntelliJ	The program is functioning differently than how it was envisioned in Astah	Ameya Mahankal
Website support and maintenance.	5	2	10	Supporting the server hosting the website and updating info regarding the clients business.	The information provided in the website does not correspond to the new updated one.	Radoslav Kiryazov
Client dissatisfaction with usability.	1	5	5	Satisfying the requests of client, providing regular updates and performing tests with employees.	Client employees unable to operate the program causing problems between the team an the client.	Joan Tammo , Zsolt Nóvé

8. Sources of Information

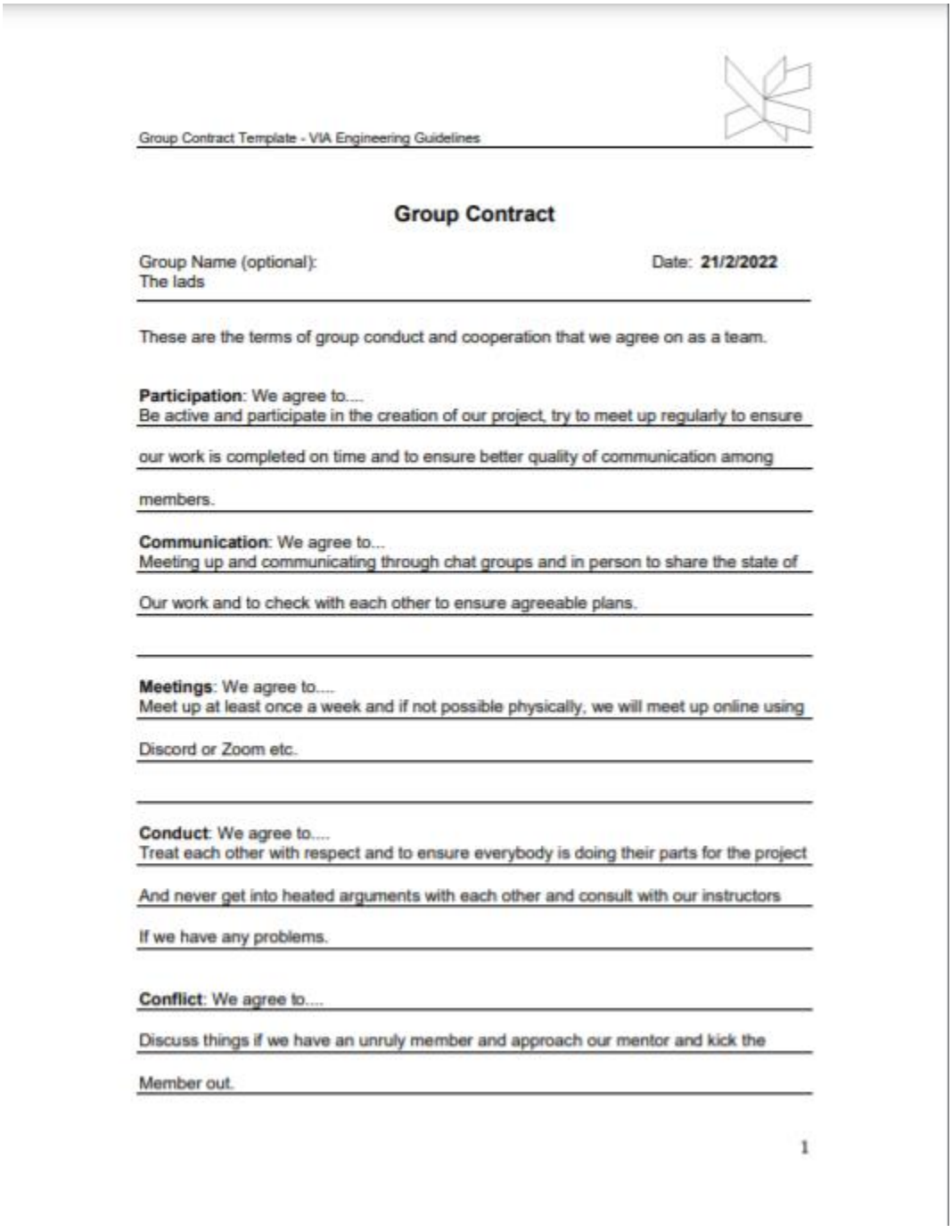
Jon Duckett. 2011. HTML & CSS Design and Build Websites. Tony Gaddis. 2015. Starting out with Java-Early Objects. 5th ed. Haywood Community College

Kenneth H. 2012. Rosen. Discrete Mathematics and its Applications. 7th ed. Monmouth University

Jon Duckett. 2011. HTML & CSS Design and Build Websites.

Appendix:

Our contract



Group Contract Template - VIA Engineering Guidelines

Group Contract

Group Name (optional):
The lads

Date: **21/2/2022**

These are the terms of group conduct and cooperation that we agree on as a team.

Participation: We agree to....
Be active and participate in the creation of our project, try to meet up regularly to ensure our work is completed on time and to ensure better quality of communication among members.

Communication: We agree to...
Meeting up and communicating through chat groups and in person to share the state of Our work and to check with each other to ensure agreeable plans.

Meetings: We agree to....
Meet up at least once a week and if not possible physically, we will meet up online using Discord or Zoom etc.

Conduct: We agree to....
Treat each other with respect and to ensure everybody is doing their parts for the project And never get into heated arguments with each other and consult with our instructors If we have any problems.

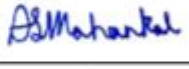
Conflict: We agree to....
Discuss things if we have an unruly member and approach our mentor and kick the Member out.

1

Deadlines: We agree to....

Ensure everyone is following through with their assigned work in order to ensure the Project is completed well before the deadline and leave nothing to the last second.

Other Issues:

Group member's name	Student number	Signature
Ameya Sundeep Mahankal	326157	
Bozhidar Ganchev Manev	326391	
Joan Tammo	325753	
Radoslav Kostov Kiryazov	326155	
Zsolt Nóvé	326345