COMPUTER SCIENCE

DAT 20I

ROSKILDE DAYCARE PROJECT

27/03/2021

By Krishna Prasad Khanal

Omar Said Farah

Mahfuzur Rahman Shawon

Table of Contents

" A picture is worth a thousand words. An interface is worth a thousand pictures" Ben Schneiderman

ABSTRACT

The success of any organization such as Roskilde Daycare Institute depends on its ability to acquire accurate and timely data about its operations, to manage this data effectively, and to use it to analyze and guide its activities. The Daycare IT-Administration system offer Administrator (like Sandra) with a unified view of data from multiple sources. The main objective of this project is to build a kid database system that will store records of kids. It is purposed to reduce time spent on administrative tasks. The system is intended to accept process, generate child name, contact number and parent information's accurately. The system is also intended to provide better services to Administrator, provide meaningful, consistent, timely data, information and finally promotes efficiency by converting paper processes to electronic form.

INTRODUCTION

Savvy IT-solutions has been contacted by Roskilde Daycare, located 30 km west of Copenhagen on the Danish island of Zealand.

Roskilde Daycare is private institution, thus has no coupling with Community's network system as other public care institutions adhere to. Hence the need for their own IT-System as they have got full responsibilities for maintaining & managing daycare's administrative tasks.

In the existing system, there was a high chance of misinterpretation of data as well as occurrence of errors. Moreover, it was cumbersome and time consuming.

Therefore, the Roskilde Daycare Management decided that it is time for an efficient IT administration system with better data storage, maintenance mechanism.

1. STATE OF THE PROBLEM

The problem of using a manual administration IT-system in Roskilde Day Care, there was a high chance of misinterpretation of data as well as occurrence of errors.

It resulted in the inability of the staff (specially Administrators) to process documents quickly, and the difficulty in achieving on accurate results.

Moreover, it was cumbersome and time consuming. With the increase in volume of registrations in the Daycare institutes, traditional method of management has gone out of phase. As a result of this, an advanced IT administration System has been the demand of time.

Hence, the above-mentioned problems prompted for the Design and implementation of new IT administration system, to enable the institution to work more effectively.

2. PURPOSE OF THE PROJECT

The purpose of this study is to design and implement a system that will eliminate or at least facilitate above-listed problems such as misplacement of important reported cases, documents and records, duplication of efforts and a lot of time that is required or taken when searching and processing of cases and files.

It is also created to expose the lapses of the existing system and to explore the intricacies associated with software design.

3. AIMS AND OBJECTIVES

The aims and objectives of this project is to ascertain on how the Day Care institution operations and other activities are performed, and also to detect problems that pose obstacles with a view of modifying the operations and developing a new computerized system that will be more effective and accurate such as in the area of misplacement of vital documents, a Database will be used to record, store and retrieve large volume of information, which will deduce the duplications of efforts due to inconsistency in activities and time taken in search of file when it is required for processing.

A database is one of the most acceptable business software application today. It is a collection of related information that is organized for case of reference.

This project describes how all these could be done electronically achieved. It shows also how information could be stored, modified and recalled instantly and accurately.

4. SCOPE, LIMITATIONS & AGREEMENT

Roskilde Daycare IT administration system will be a web-based application as per the agreement with our clients (Mr. Janus Pederson, Mr. Douglas Beaver, and Mr. Kristoffer Michael).
In making the online registration, administrators must be able to quickly
register kids details into the system. They should be able to search or fetch client details without any problem. And lastly, but not least, can make some pre-

schedule daily events like say at 8:30 is ‘Breakfast’ time, at 10:40 is outdoor play & 12:30 ‘Lunch’ time...etc.
Although It's a web-based Management system, it is not designed for other external users (such as parents or kids for that matter on the internet) from interacting with it, Our system will *not* tie into employee (i.e., Admin) payroll, time management system, or into parent login capability & information. This is mainly since, in this project, there's no true systems that are already in place. We would have to create the "existing" systems to tie into.

MAIN SECTION: ORGANIZATION

STAKEHOLDERS

Identifying the most important Stakeholders:

What the Roskilde Daycare board (i.e., management) wants to see in the future?

- Kids Number Growth
- Good Rapport with Parents/Guardians
- Economy Growth.

What is the expectation of the management from the new System?

- To be easier to register new kids.
- Easy to make an Appointment
- High Productivity
- System developers can meet the deadline.
- No renewal of (new) System for the next five years.

What will the management does not want to see in the future?

- Poor rapport/relationship with Parents/Guardians
- Bad System
- Decreasing Kids Number
- Declining turnover.

Stakeholder analysis

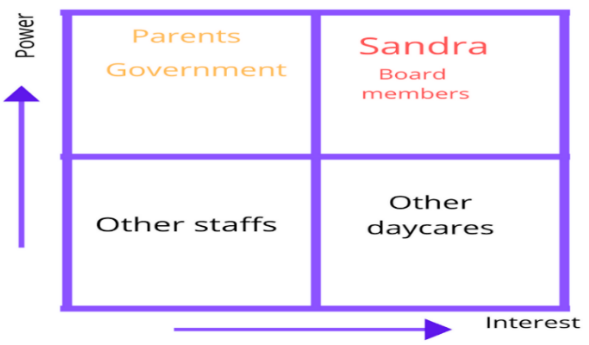
SN	Stakeholders	Key Interests	Power
1.	Sandra Madsen	She is the main stakeholder who will use our system most frequently, she needs this system that is comfortable, reliable and productive.	If she is not satisfied our project even can be cancelled.
2.	Other workers	These are also interested in our product; they are also used frequently.	They also need to be satisfied with our system; consequences can affect our project.
3.	Parents	Parents will be interested indirectly to the system if their data handled properly or not, personal data are secured or not?	We need to watch them carefully though there is no direct impact but only indirect influence.
4.	Board members	Board members have a direct interest in our system, they will be interested in the data and documents our system manages.	They need to be satisfied they are as powerful as Sandra Madsen
5.	Government	Government and other law enforcement would be indirectly interested in our project.	We must keep satisfied in such a way that our system does not violate the rules and regulations.

6.	Rival day-care	They may be of interest to our system indirectly	They are not powerful, but we cannot neglect them watch them carefully.

Risk analysis

Sn	**Risk factor**	**Mitigation**
1.	A team member can be sick.	In this situation, we are prepared to take over the work of sick members and divide the work to the rest of the members.
2.	Loss of data	To avoid the loss of data because of disc failure or other reasons we will use GitHub and google drive as a backup.
3.	System failure or any hack	We will use the good antivirus and create the system restoring point in the present machine.
4.	Fire or water damage of physical documents and resources provided by day-care	We will scan all the documents provided by day-care and put in drobox so that it is accessible and safe.
5.	New unfavorable rules from governments.	We will consult a good legal advisor.

Stakeholder Grid



SWOT Analysis:

STRENGTH

- Daycare opens from 6 am to 6 pm so that more service time.
- Centrally located
- Good playgrounds and equipment
- Providing service for many years so that most of the local parents know it.

WEAKNESS

- No digital documentation yet
- Lack of networking between other daycare
- No online presence
- Less available places for children causing long waiting list.

OPPORTUNITY

- Creating network between others daycares
- Developing online presence
- Generating more free places so that more children can get a place.
- Expansion of indoor playgrounds so that kids can play even more in winter time

THREATS

- Poor administrative system and document management.
- New government rules
- Data handling of children and parents is not addressing GDPR regulations.
- New diseases like COVID 19

FURPS

Users of the System

1. **Admin** (Employees)

Functional Requirements

In short, these requirements emphasis the Features of the Application.

A good question is: What *must* this Application/Software System *do*?

1. Admin

1. Can login and logout
2. Can add kid details
3. Can view kid details
4. Can delete kid
5. Can update kid details
6. Can make an Appointment
7. Can arrange a Schedule

Non-Functional Requirements

Since the non-functional focus more on Characteristics instead of Features, question that can be

Asked could be: *How* the Application / Software System *should be*?

availability

Browser support

MAIN SECTION: SOFTWARE DEVELOPMENT (SWD)

Group Contract

As group contract is to promote academic integrity among all members, every member must meet the contract or face the consequences decided upon by the group. Agreement is only as good as the consequences of breaking agreements.

Every member should be able to fulfil its fair part of the project and add some value to the team & product itself.

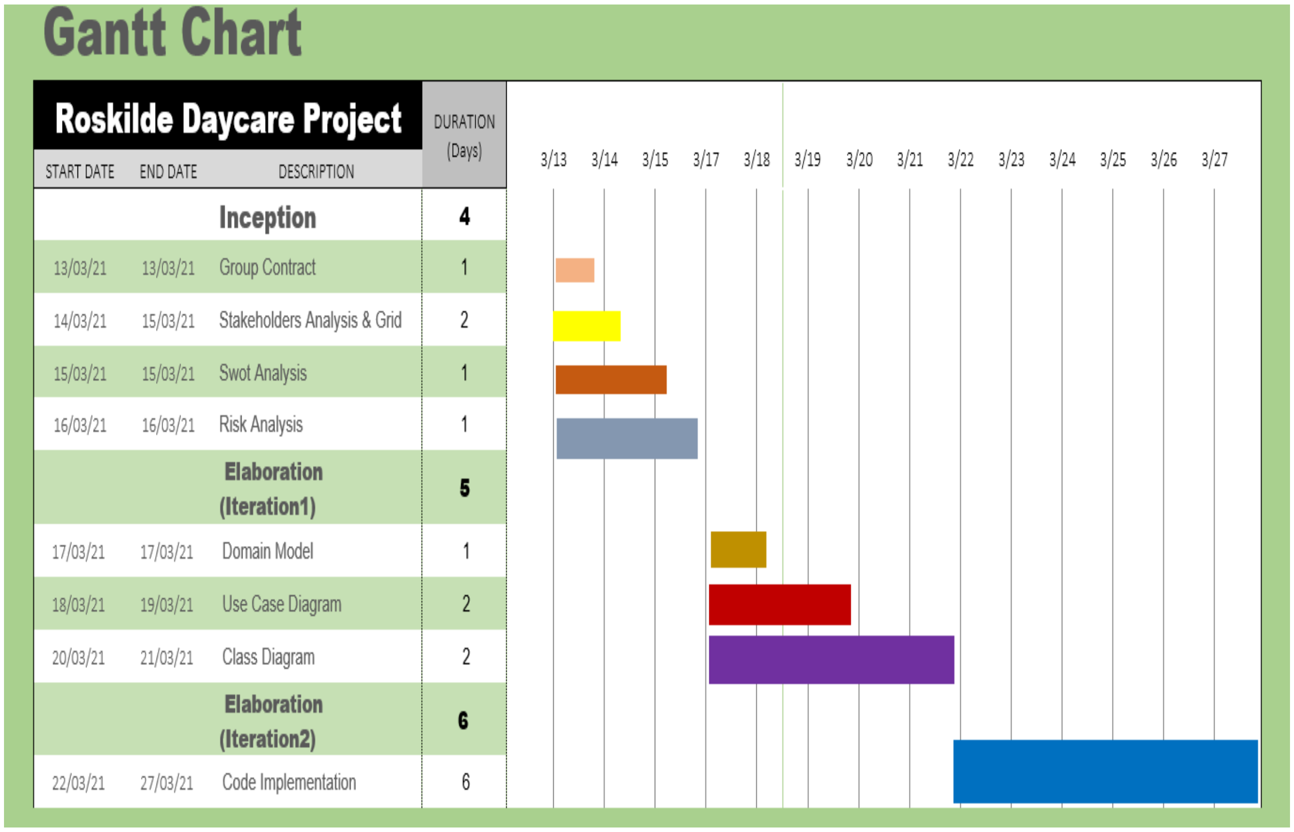
Team Agreement:

- Each member should accomplish any task assigned to them by the Group

- In case of absence, member will check-in with Group & strive to fulfil its parts
- Each member should meet the deadline for the project or set by the Group
- Member should be open to constructive criticism by another Group member

And consequences for violating & conflict management should be dealt with appropriately too.

Team Member's Name
-Krishna Prasad Khanal
-Omar Said Farah
-Mahfuzur Rahman Shawon



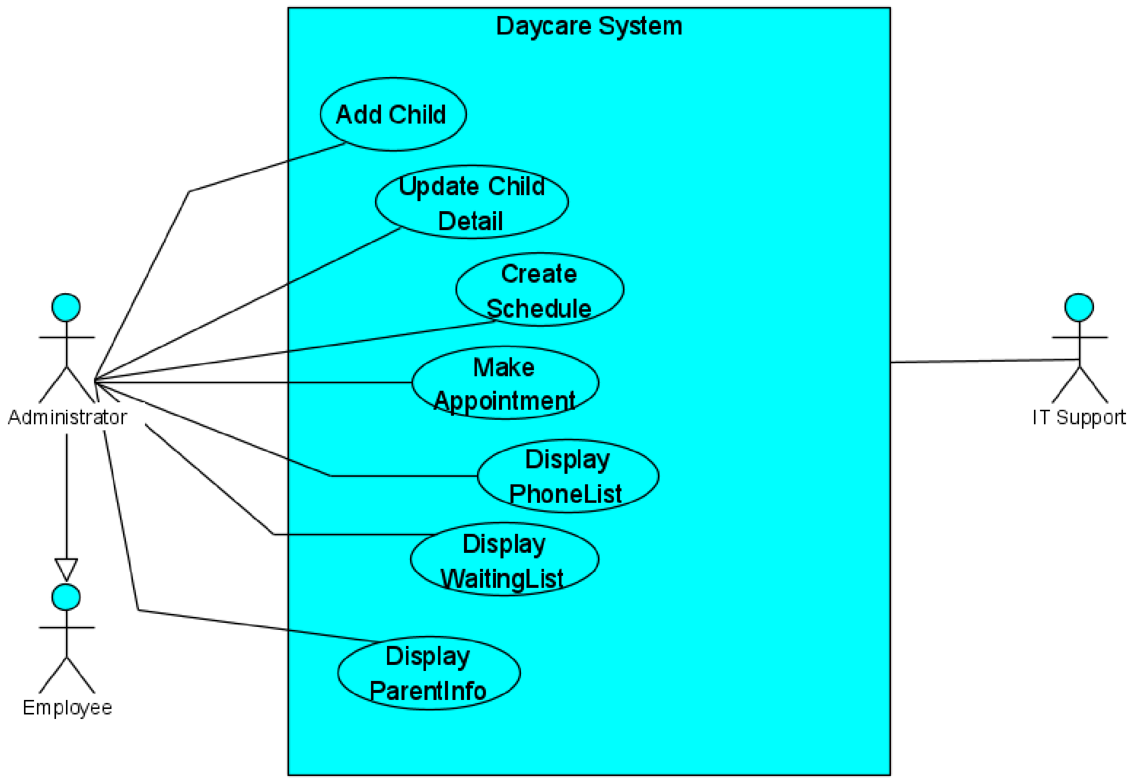
Gantt Chart

Roskilde Daycare Project			DURATION (Days)
START DATE	END DATE	DESCRIPTION	
		Inception	**4**
13/03/21	13/03/21	Group Contract	1
14/03/21	15/03/21	Stakeholders Analysis & Grid	2
15/03/21	15/03/21	Swot Analysis	1
16/03/21	16/03/21	Risk Analysis	1
		Elaboration (Iteration1)	**5**
17/03/21	17/03/21	Domain Model	1
18/03/21	19/03/21	Use Case Diagram	2
20/03/21	21/03/21	Class Diagram	2
		Elaboration (Iteration2)	**6**
22/03/21	27/03/21	Code Implementation	6

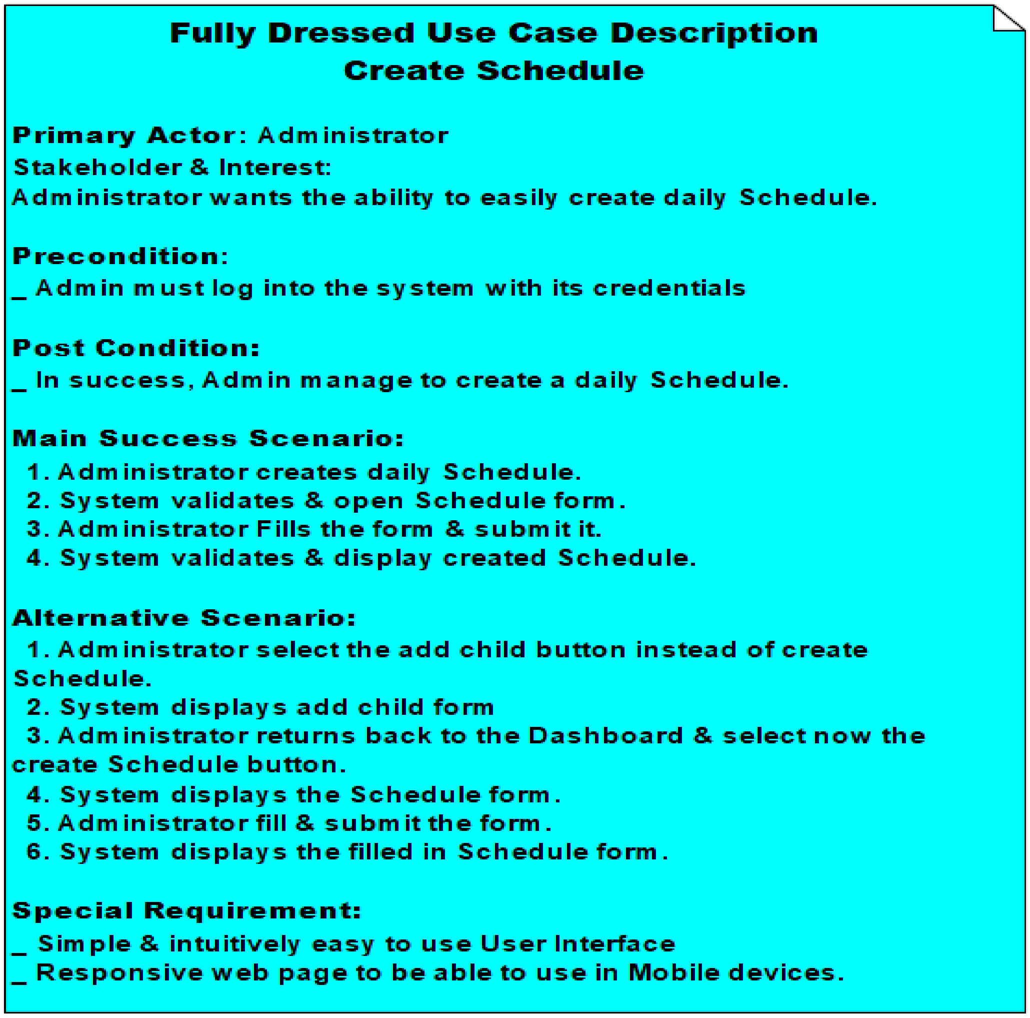
11.USE CASE DIAGRAM

The section below outlines the Use Case Diagram/ behaviour Diagram for the Daycare System. A Use Case is a diagram of several Use Cases & multiple Actors at the same time, as we can see clearly below. We have got Use Case diagram & written Use Cases which both show different perspectives and complement one another. As general rule of thumb, we started by listing Use Cases first before drawing anything in Visual Paradigm. It helps tremendously to ask ourselves, in a case of a particular Use Case, question like: is it a goal that Actor want to accomplish?

Use Cases describe a set of actions/ tasks that the system should or can perform in collaboration with one or more external users of the system (actors). For each Use Case, the precondition and post-condition have been articulated along with Actors and the associations/ relationships (internal/ external) have been clearly stated. In short, Use Case allow Actors to accomplish a Goal, thus Use Case should be a Goal itself not part of it. Therefore, as we can see below, we can safely say we have seven goals that our Actors (i.e., Administrators like Sandra) should accomplish while interacting with our System. According to our requirement, our Actors are role-based (i.e., Admin).



The Create Schedule is a Fully Dressed Use Case Description, and as such, we filled all required field for a casual description plus some more.



Fully Dressed Use Case Description
Create Schedule

Primary Actor: Administrator
Stakeholder & Interest:
Administrator wants the ability to easily create daily Schedule.

Precondition:
_ Admin must log into the system with its credentials

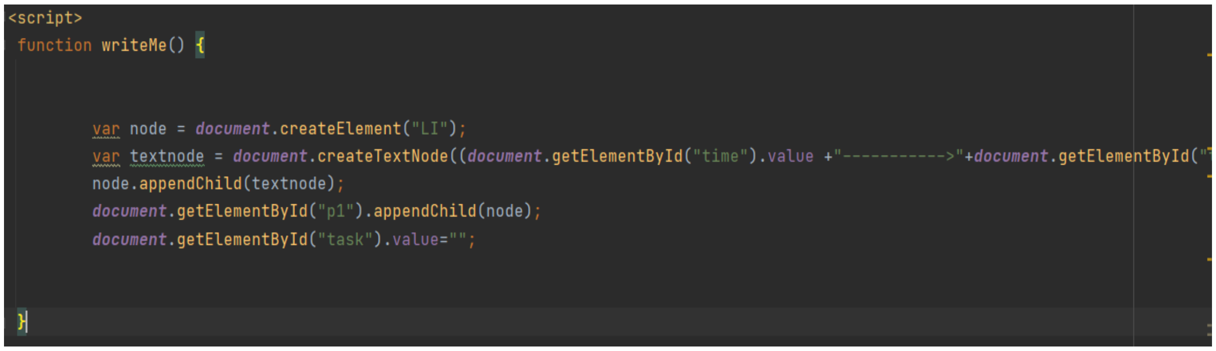
Post Condition:
_ In success, Admin manage to create a daily Schedule.

Main Success Scenario:
1. Administrator creates daily Schedule.
2. System validates & open Schedule form.
3. Administrator Fills the form & submit it.
4. System validates & display created Schedule.

Alternative Scenario:
1. Administrator select the add child button instead of create Schedule.
2. System displays add child form
3. Administrator returns back to the Dashboard & select now the create Schedule button.
4. System displays the Schedule form.
5. Administrator fill & submit the form.
6. System displays the filled in Schedule form.

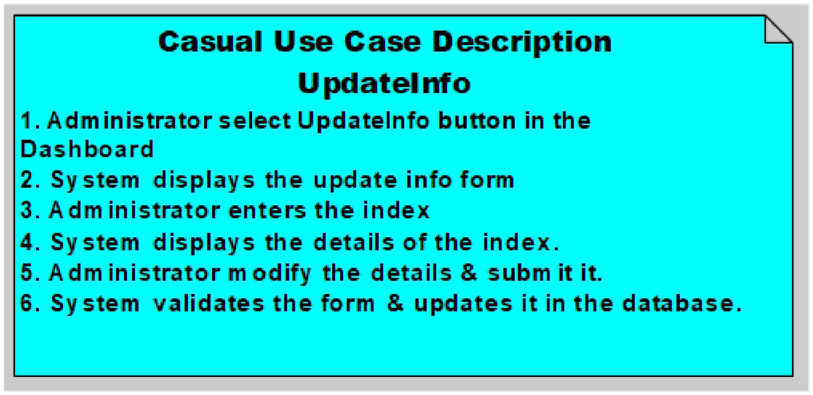
Special Requirement:
_ Simple & intuitively easy to use User Interface
_ Responsive web page to be able to use in Mobile devices.

Below is the script responsible for the above fully dressed Create Schedule Use Case.



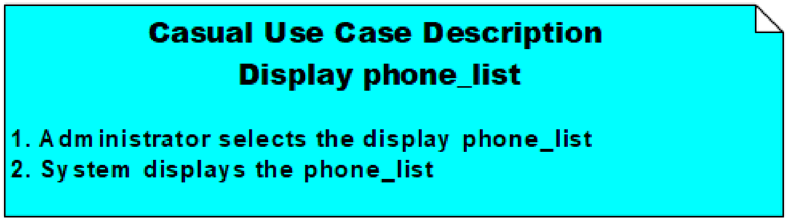
```
<script>
 function writeMe() {


        var node = document.createElement("LI");
        var textnode = document.createTextNode((document.getElementById("time").value +"----------->"+document.getElementById("
        node.appendChild(textnode);
        document.getElementById("p1").appendChild(node);
        document.getElementById("task").value="";


 }
```

Below we have 4 casual Use Case descriptions: update info, display phone list, display waiting list & display parent info.



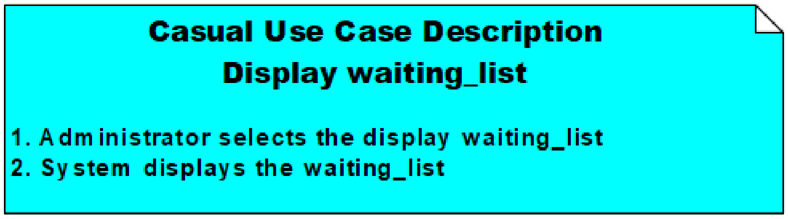
Casual Use Case Description
UpdateInfo

1. Administrator select UpdateInfo button in the Dashboard
2. System displays the update info form
3. Administrator enters the index
4. System displays the details of the index.
5. Administrator modify the details & submit it.
6. System validates the form & updates it in the database.



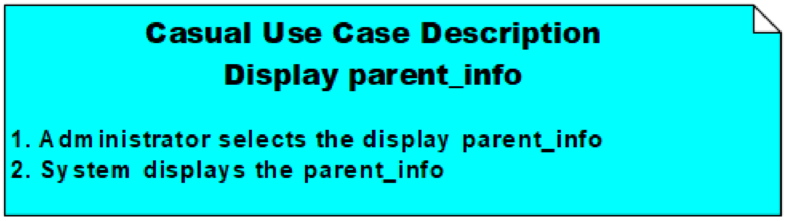
Casual Use Case Description
Display phone_list

1. Administrator selects the display phone_list
2. System displays the phone_list



Casual Use Case Description
Display waiting_list

1. Administrator selects the display waiting_list
2. System displays the waiting_list



Casual Use Case Description
Display parent_info

1. Administrator selects the display parent_info
2. System displays the parent_info

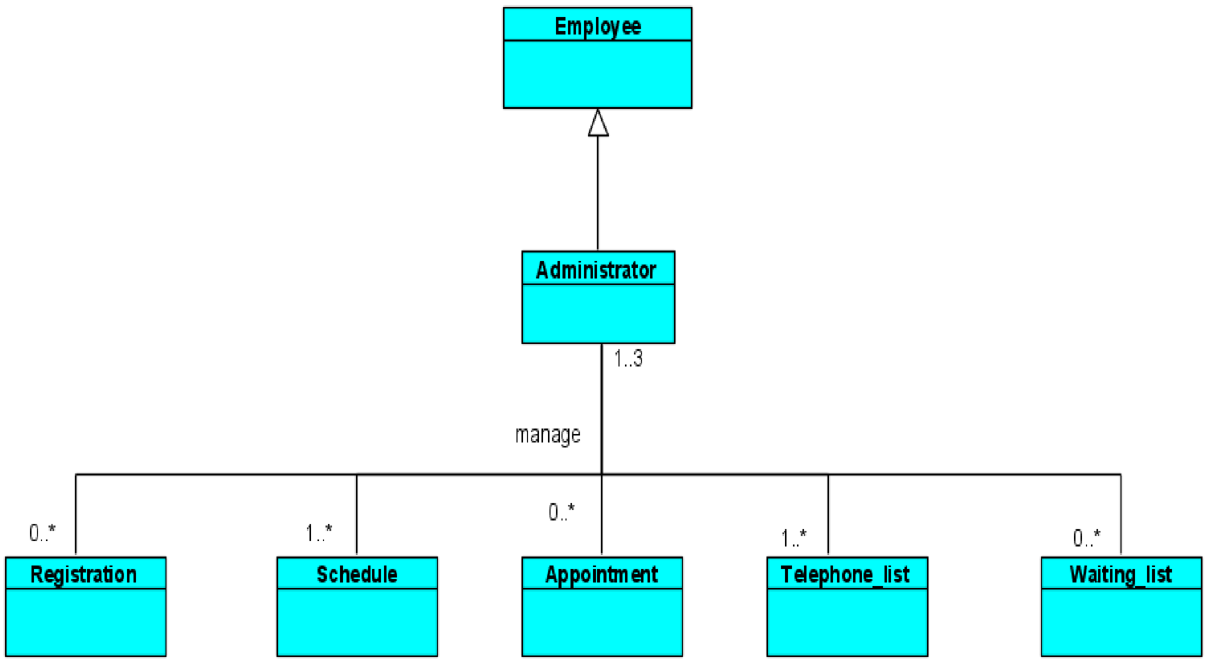
12.DOMAIN MODEL

After defining the requirements and writing some Use Cases, we start to transition from Analysis (understanding the problem we're trying to solve) to Design (how we're going to organize our solution). In our case, here is a Conceptual Model for the Daycare System describing its different components & their interactions (how they are related to each other). The best practice for creating a Conceptual Model is to first outline all the nouns from the requirements stand point. Here is short cut out from it:

Examples of administrative tasks are; create a work schedule, a telephone list, a waiting list for
new children, parent information, etc.

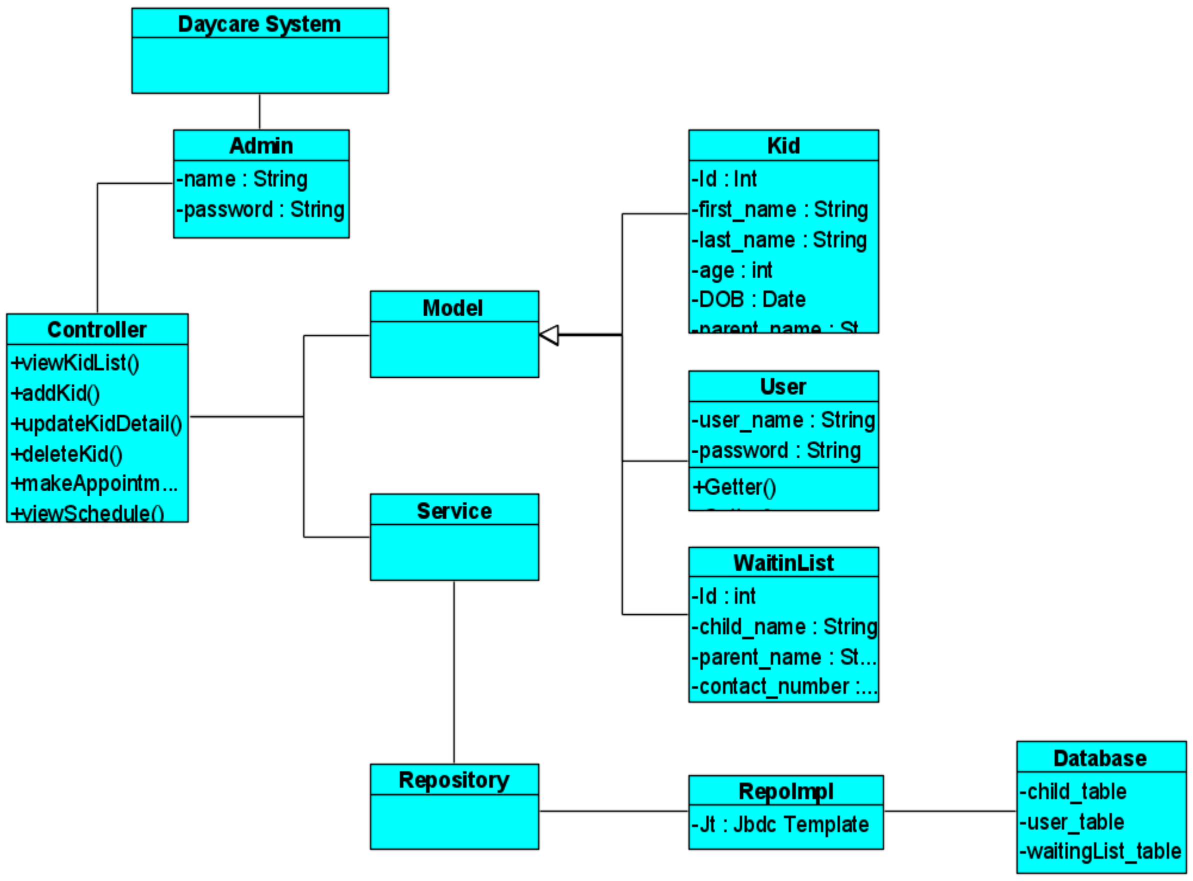
Now let's just list all the nouns: schedule, telephone list, waiting list, new children, parent…etc.

We all know that not all of them (i.e., list) will be object or class but they are definitely good candidate starting point for our Domain Model at our current stage.



13.CLASS DIAGRAM

Class Diagram is the most important UML diagram used and help construct the code for Software Application Development. As our implementation is based on Java Spring, it is not a surprise that we'll depict them in that manner. We are at the last stage of transitioning from Design to Implementation. Hence, it is much easier to translate Class Diagrams into code.



MAIN SECTION: SOFTWARE CONSTRUCTION (SWC)

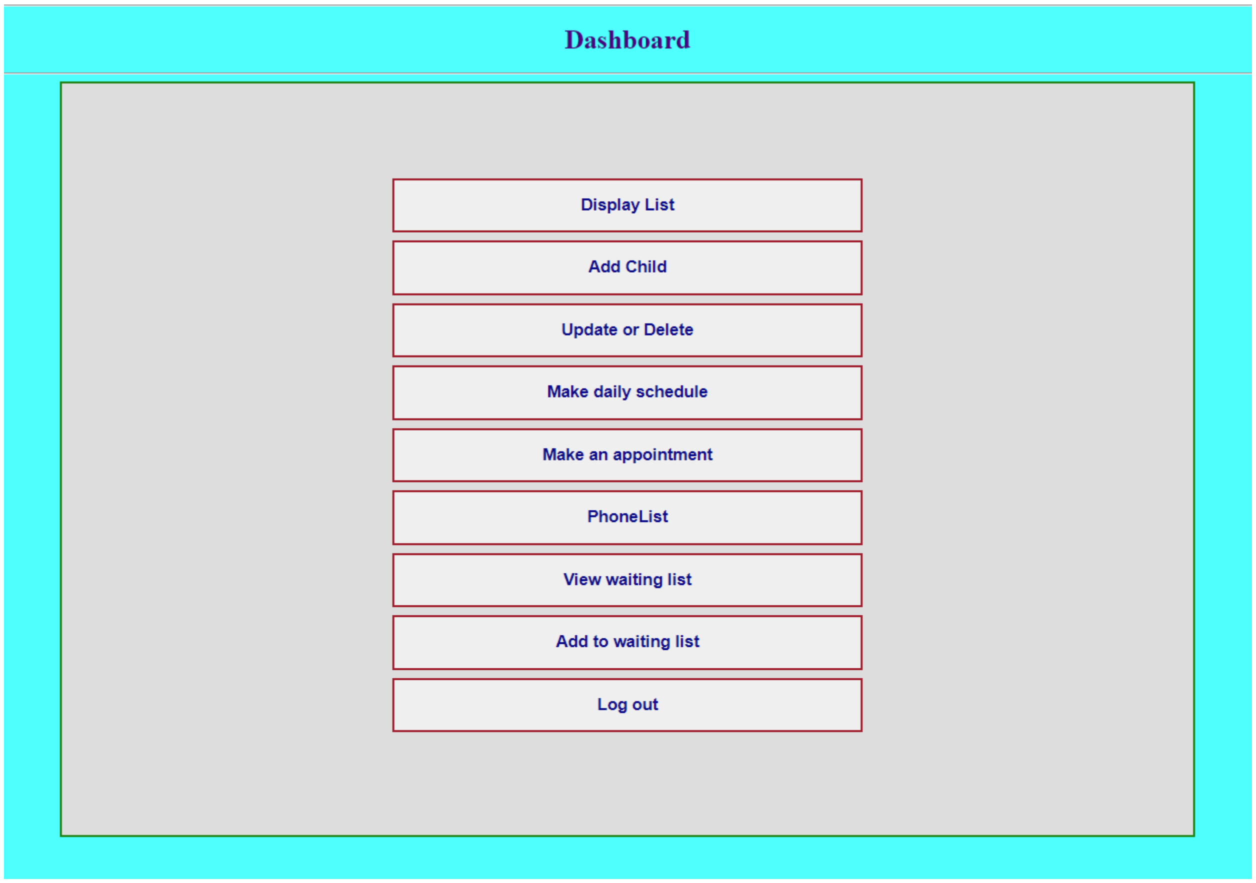
The aim in this section is to showcase the Programming process, the way to the Implementation.

As per agreement with our clients, the application system is web-based and as such we opted for the usage of Java Spring Framework. Technology used are: Spring Boot, IntelliJ Idea IDE, MySQL database and Apache Maven for build tools of choice.

For the web presentation, we are following the MVC design patterns for better layered approach. As it hid our business logic by only to view what is necessary through the use the single point of contact with the Controller.

Finished Product Preview

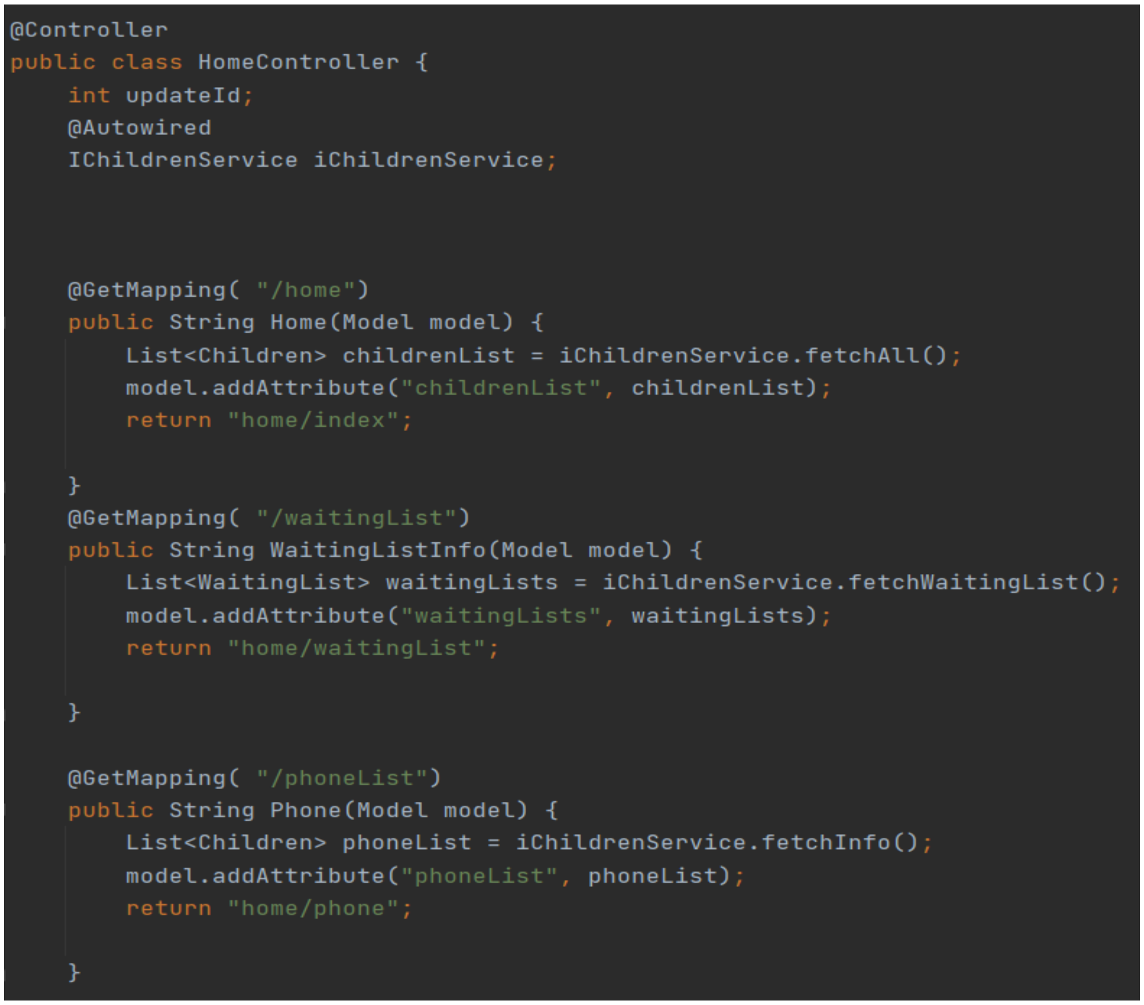
Here is our Welcome (i.e., Dashboard) page where all of the features of our application are displayed in accordance to the requirements.



Design to Implementation

During coding and implementation, we will create the actual product i.e., write/code the set of programs to render business functionality as per Software Design Pattern.

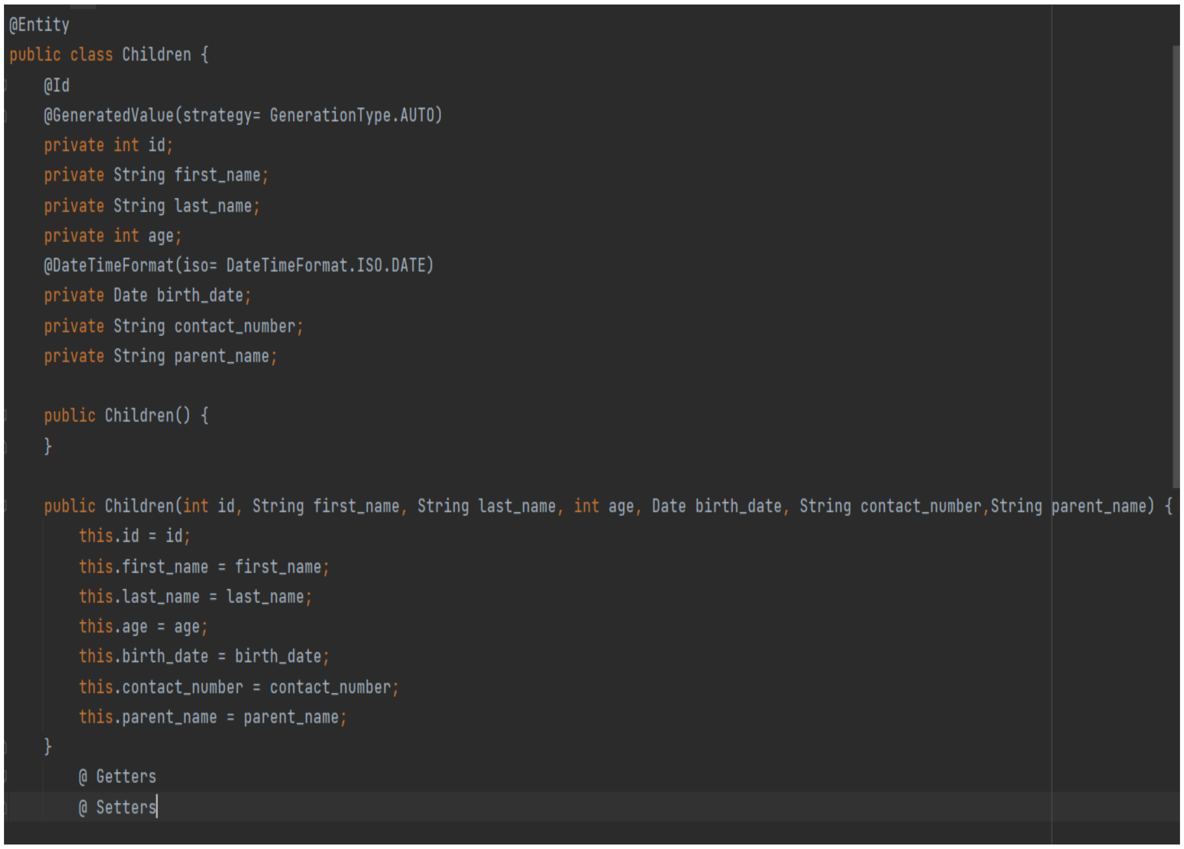
As we opted to use Spring Framework not only because of its lightweight but also for its well-known ease of providing both Standalone & Enterprise-grade Application.
We followed the MVC design pattern where we used Spring Web
& Thymeleaf alongside Apache Maven as build tools of choice. MySQL database being our backend server for optimal productivity, scalability & performance.
The output of the coding is the source code for the Software.
In the following section, we will try to showcase some snippets of our code for presenting the bigger picture & show the transitioning from Design to Implementation.
As we already touched on previously, our code is adhering to the MVC Design Pattern we will thereby start with the Controller (I.e., home Controller):
Here are some snippets of our implementation (i.e., source code). We start with the Controller (C part of MVC Design Pattern), it contains the business logic by handling the requests, storing & retrieving data & placing data into model.



```
@Controller
public class HomeController {
    int updateId;
    @Autowired
    IChildrenService iChildrenService;



    @GetMapping( "/home")
    public String Home(Model model) {
        List<Children> childrenList = iChildrenService.fetchAll();
        model.addAttribute("childrenList", childrenList);
        return "home/index";

    }
    @GetMapping( "/waitingList")
    public String WaitingListInfo(Model model) {
        List<WaitingList> waitingLists = iChildrenService.fetchWaitingList();
        model.addAttribute("waitingLists", waitingLists);
        return "home/waitingList";

    }

    @GetMapping( "/phoneList")
    public String Phone(Model model) {
        List<Children> phoneList = iChildrenService.fetchInfo();
        model.addAttribute("phoneList", phoneList);
        return "home/phone";

    }
```

We can clearly see our home, waiting list & phone list endpoints where the Home Controller handles to select the view, collect the data (i.e., model) and wires both together to send for the appropriate view template.

Now, as for the data, here is a snippet of our model, being children class Entity.



```
@Entity
public class Children {
    @Id
    @GeneratedValue(strategy= GenerationType.AUTO)
    private int id;
    private String first_name;
    private String last_name;
    private int age;
    @DateTimeFormat(iso= DateTimeFormat.ISO.DATE)
    private Date birth_date;
    private String contact_number;
    private String parent_name;

    public Children() {
    }

    public Children(int id, String first_name, String last_name, int age, Date birth_date, String contact_number,String parent_name) {
        this.id = id;
        this.first_name = first_name;
        this.last_name = last_name;
        this.age = age;
        this.birth_date = birth_date;
        this.contact_number = contact_number;
        this.parent_name = parent_name;
    }
        @ Getters
        @ Setters
```

The above code will be what is mapped or mirrored in our Database.

CONCLUSION

Information is an indispensable tool many organizations use to advance decision making. Large number of children's data are generated either manually or electronically on daily basis.

When population of children in a school is less than a hundred, the manual system may seem to work perfectly but it is not the best method of managing records of children. The manual and disintegrated electronic systems have numerous disadvantages because these methods of capturing and managing data about children are prone to data inconsistency, data redundancy, difficult to update and maintain data, (not to include) bad security, difficult to impose constraint on various data file and difficult to backup. An integrated children database system provides prudent solutions to address problems associated with manual system. In order to manage children overtime, there is the need to use past records of children without any missing data. The database system which captures and maintains longitudinal data of kids would provide an accurate and reliable data about current and past children. The system is free of errors and very efficient and less time consuming due to the care taken to develop it. All the phases of software development cycle are employed (see Gantt Chart) and it is worthwhile to state that the system is very robust.

We feel that the risks of this project do not pose any unsolvable threats to the time restraints of this project. We have access to the necessary technology to successfully create and develop our software.

Provision is made for future development in the system, having web-based Application System integrated with MySQL DB, where it will be easier to introduce more features in the System such as Food Program record, Photo Gallery, Billing management, Parent Update, Report Management…etc. We have developed with the future in mind.

Recommendation

Since children database system is very broad, the scope of this project covers only a small aspect of children information system due to the fact that the stipulated two weeks within which the project is expected to be executed is too short.

This report could be useful to any person who wants to do a project on similar topic.

Opportunity & Lesson Learned

During the course of this project, the research team was able to understand better what goes in the child's records management system in a Daycare institution. This was effectively done through reading of literature of our Curriculum and research. The whole process of developing the system was an opportunistic challenge. Seeing the system into a tangible system was a rewarding exercise.

BIBLIOGRAPHY

The whole group
Books:

- Craig Larman: Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development - Third edition (2004)
- Murach's MySQL (3ed)
- Building Java Programs
- The Organization

search the Google machine.