Lebanese University
الجـــامعة اللبنـــانية

OBAL
MEDICAL LABORATORY

Under the direction of: Dr. Mohammad Saade

Prepared by:

Alaa Shafshak,Mohammad Hawchar,

Zakaria Mawass

Acknowledgment

We would like to express our deepest appreciation to all those who provided us the possibility to complete this project. We have taken efforts in this project. However, it would not have been possible without the kind support and help of Dr Saade.

Chapter 1: Project Specifications

Introduction

Description

We had the opportunity to put into practice our knowledge and acquired skills to complete this project.

“OBAL” is a desktop application that aims to be used internally for Obal medical laboratory. It helps the employees achieve their daily tasks faster and with more accuracy.

Our application saves time by handling patients’ data management to enhance the quality of services and the chance to grow by time.

“OBAL” offers the possibility to request for orders, print results as well as printing bills.

Functional requirements

A functional requirement specifies the action a system must be able t to perform, excluding physical constraint: a requirement specifying an input / output behavior of a system.

In this context our application, mainly implements the following features:

- Adding a client: The system allows employees adding and searching for clients.
- Managing doctors: The system allows employees adding, deleting and searching for doctors.
- Requesting orders: The system allows employees requesting orders for patients to further make their analysis.
- Making results: The system allows employees generating results according to patient’s analysis.
- Printing bills: The system allows employees printing bills.

Non-Functional requirements

A non-functional need is a requirement specifying system properties, such as environment and implementation constraints, performance requirements, platform dependency, ease of maintenance, scalability, and reliability.

In our system we distinguish the following non-functional needs:

- Usability: Ergonomics of man-machine interfaces and ease of use.
- Control: Controlled input according to the predefined choices.
- Scalability: Ability to well-function and take advantage of rescaled situations.

Actors

- Secretary: The secretary can:
  - Add new clients
  - Edit Clients
  - Register analysis
- Assistant: The assistant can add analysis results.
- Patron: The patron can:
  - Manage clients lists
  - Print results
  - Print bills

Overall Description

Product Features

The employee will be able to create an account for patients and enter their data. They're going to dispose orders for patients to further make analysis. After the analyses are done, the employee will print the results and the bills.

Operating Environment

“OBAL” will operate on every pc/MAC/LINUX that has JDK pre-installed.

User Documentation

User documentation will be available upon request.

Chapter 2: Project management

Project Charter

1.0 PROJECT IDENTIFICATION	
Name	OBAL
Description	It’s a desktop application for Obal medical laboratory, that manages clients accounts and their analysis and bills
Sponsor	Lebanese University
Project Manager	Dr. Mohammad Saade
Project Team Resources	Participants(Alaa Shafshak, Mohammad Hawchar, Zakaria Mawass)

2.0 BUSINESS OBJECTIVES AND SUCCESS CRITERIA
▪ Delivers a functional, reliable, well-documented software ▪ Enhances the quality ▪ Eases employees’ work ▪ Improves data security

3.0 PROJECT OBJECTIVES
Develop a well-handed, easy-to-use application

4.0 PROJECT SCOPE
- Planning for each step - Improving student's skills and teamwork

5.0 CONSTRAINTS			
DIMENSION	**CONSTRAINT**	**DRIVER**	**DEGREE OF FREEDOM**
Features	Final product must manage customers analysis and email or print results and bills (high priority). An authentication system for the employees would be desirable but non-essential(low priority).	To focus on high priority features, and design the software such that other features can be easily added at a later date.	100% of high priority features must be included in. 60% of low priority features may be included.
Quality	Final product must be industrial strength. The quality will be constrained by the abilities of the development team.	To produce a product with as few bugs as possible.	90-95% of tests must pass.
Cost	The team will be provided; aside from that there is no further budget.	To use developer-hours wisely.	No degrees of freedom.
Schedule	The schedule outlined is fixed.	The final product must be complete by July 30, 2020.	One milestone may be delivered late, with no effect on the quality of the final product.
Staff	Maximum team size is 3. All staff have other commitments outside the project.	To assign work to the team member which can complete it most efficiently.	No degrees of freedom.

6.0 MILESTONES

EVENT OR DELIVERABLE	TARGET DATE	RESPONSIBILITY
Assemble project team	May 5, 2020	Project Manager
Project charter approved	July 6, 2020	Project Manager
Project plan completed	September 1, 2020	Project Manager and Design Lead
Project plan approved	September 3, 2020	Project Manager
System design completed	September 25, 2020	Design and Development Lead
Test plan completed	September 27, 2020	Quality Assurance and Testing Lead
Customer acceptance	September 28, 2020	Quality Assurance and Testing Lead
Project closed out	October 2, 2020	Project Manager

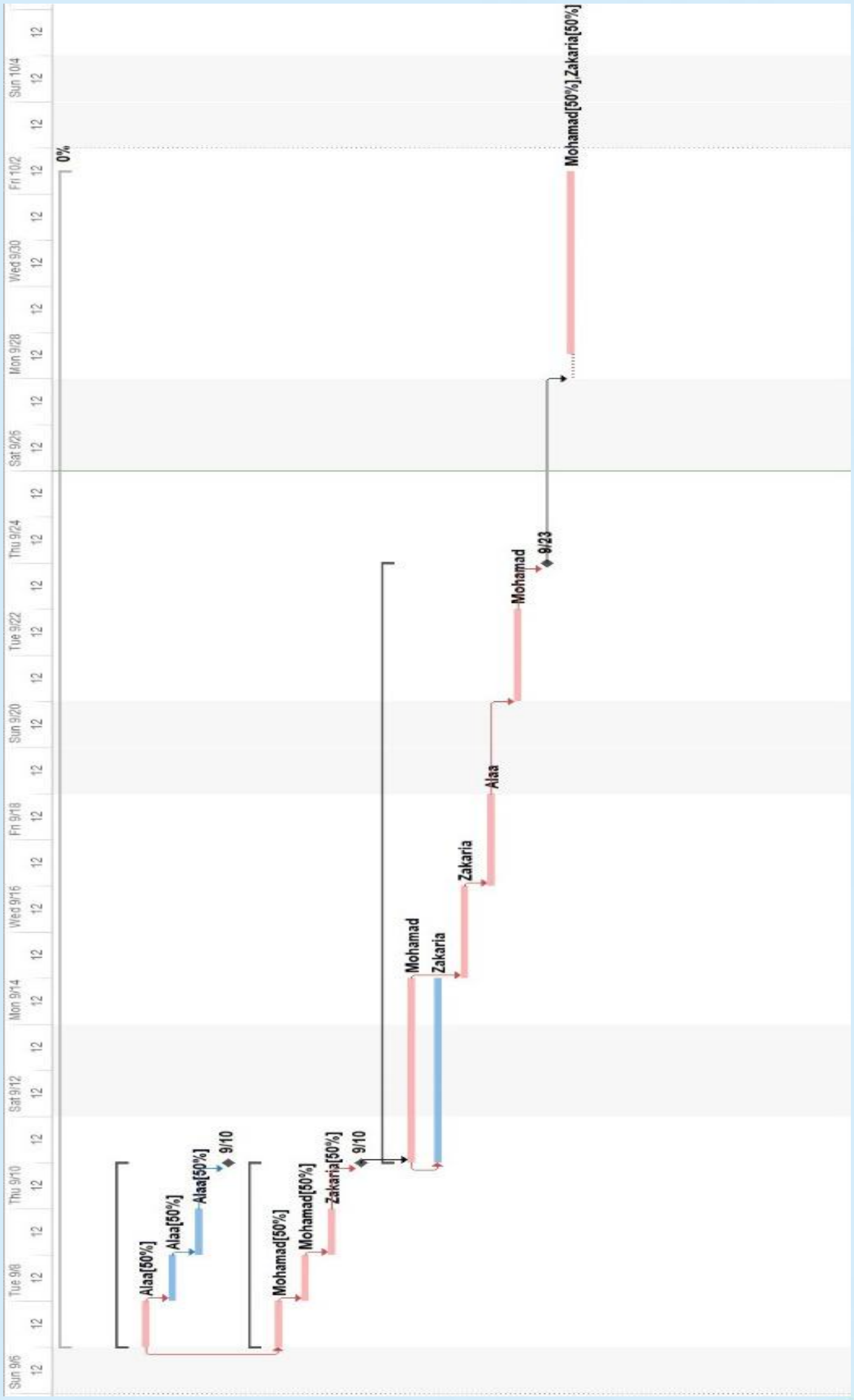
7.0 BUSINESS RISKS

RISK	PROBABILITY	IMPACT	MITIGATION
Project fails	Low	Severe	Start early and stick to an organized plan. Seek help as soon as any problems arise
Software has negative effect on system or users	Low	Medium to severe	Rigorous testing
Software poses a security risk	Medium	Severe	Limit access to resources

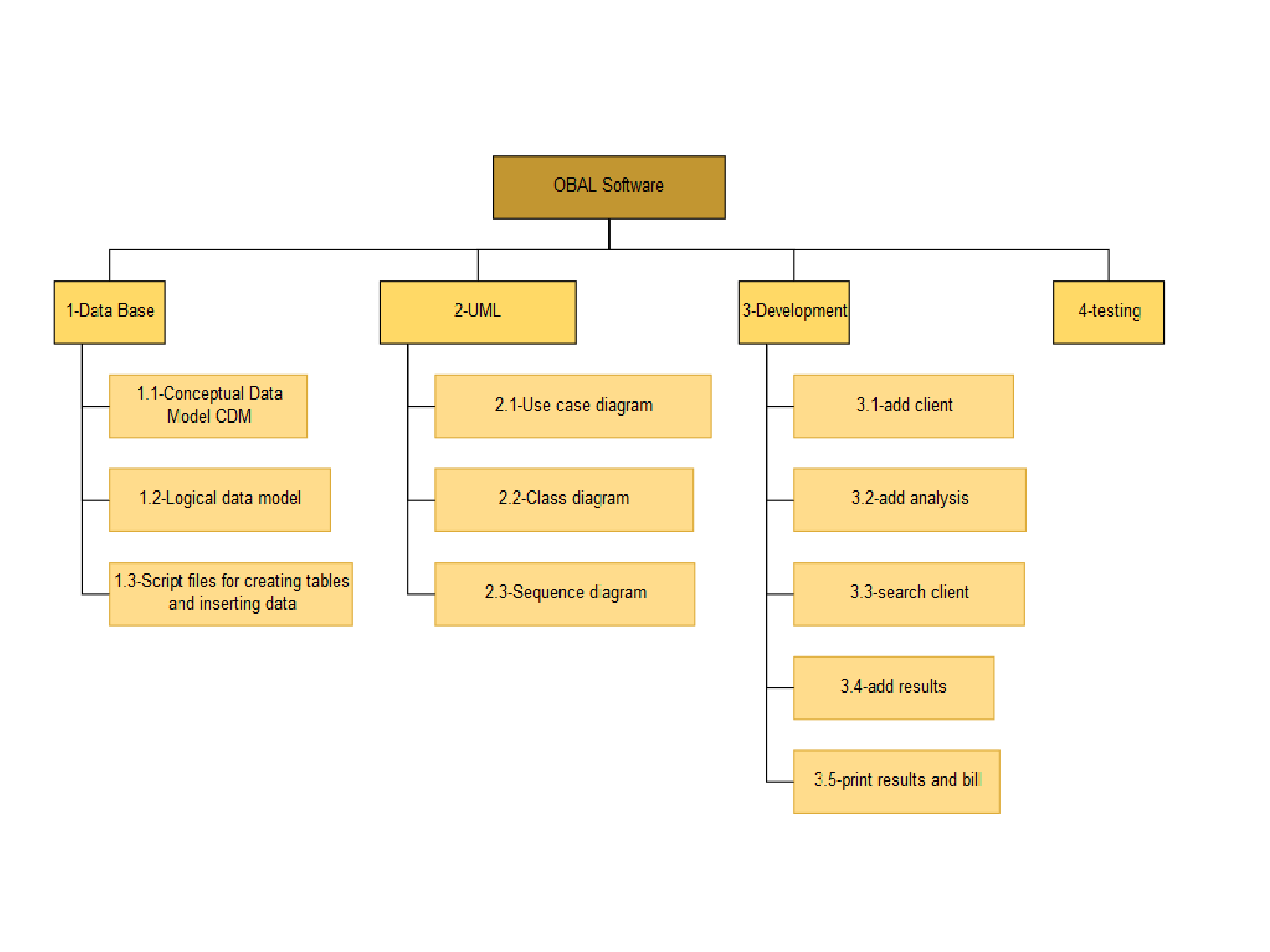
8.0 RESOURCES

RESOURCE	DESCRIPTION AND SOURCE
Development team (Mohamad Hawchar , Alaa Shafshak , Zakaria Mawass)	The development team consists of three people. Between them they will share the roles of Project Manager, Design Lead, Development Lead. Furthermore, all three will help with coding and testing.

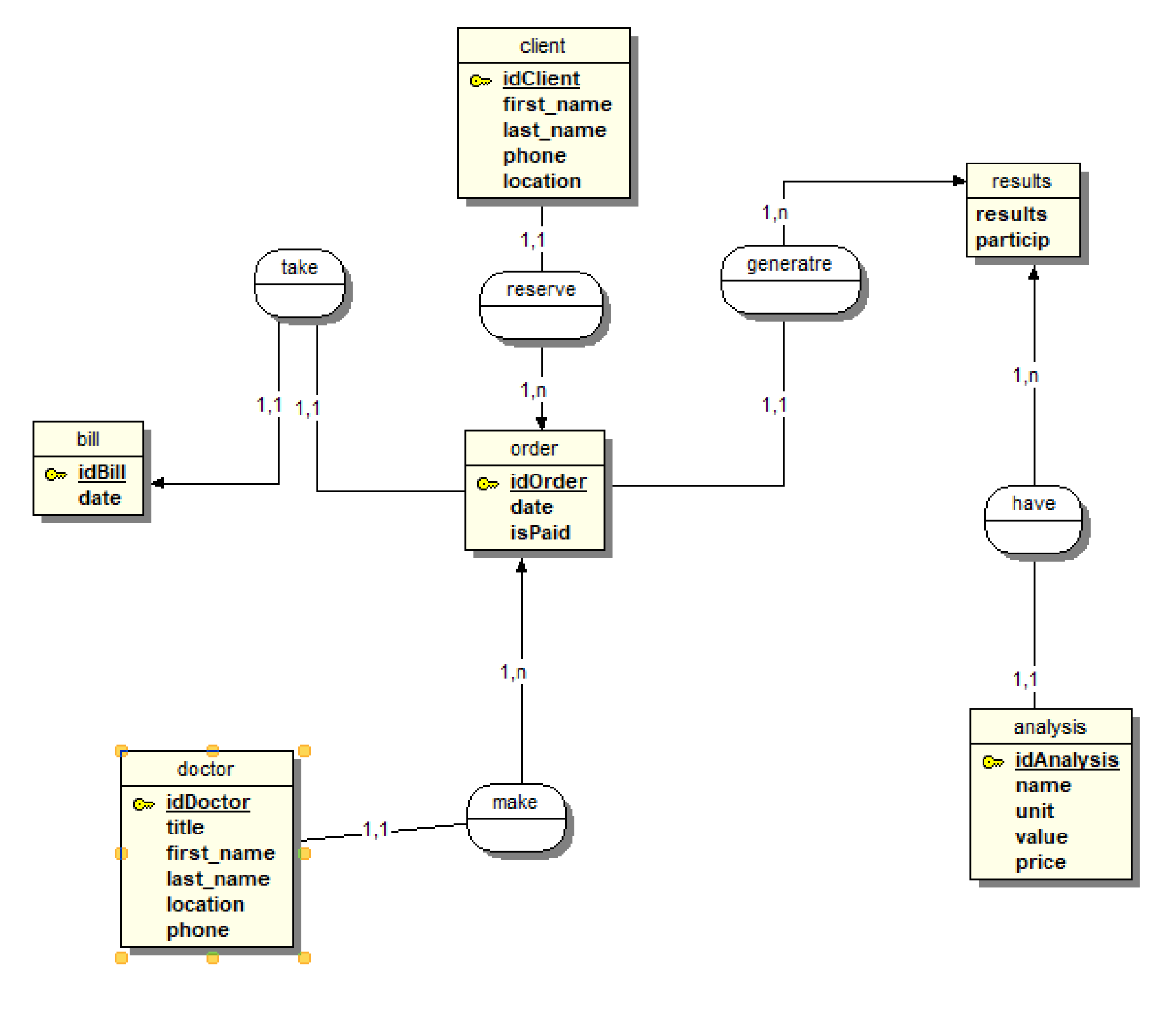
Gantt Chart


Sun 9/6
Tue 9/8
Thu 9/10
Sat 9/12
Mon 9/14
Wed 9/16
Fri 9/18
Sun 9/20
Tue 9/22
Thu 9/24
Sat 9/26
Mon 9/28
Wed 9/30
Fri 10/2
Sun 10/4
12
Alaa[50%]
Alaa[50%]
Alaa[50%]
9/10
Mohamad[50%]
Mohamad[50%]
Zakaria[50%]
9/10
Mohamad
Zakaria
Zakaria
Alaa
Mohamad
9/23
Mohamad[50%],Zakaria[50%]
0%

WBS

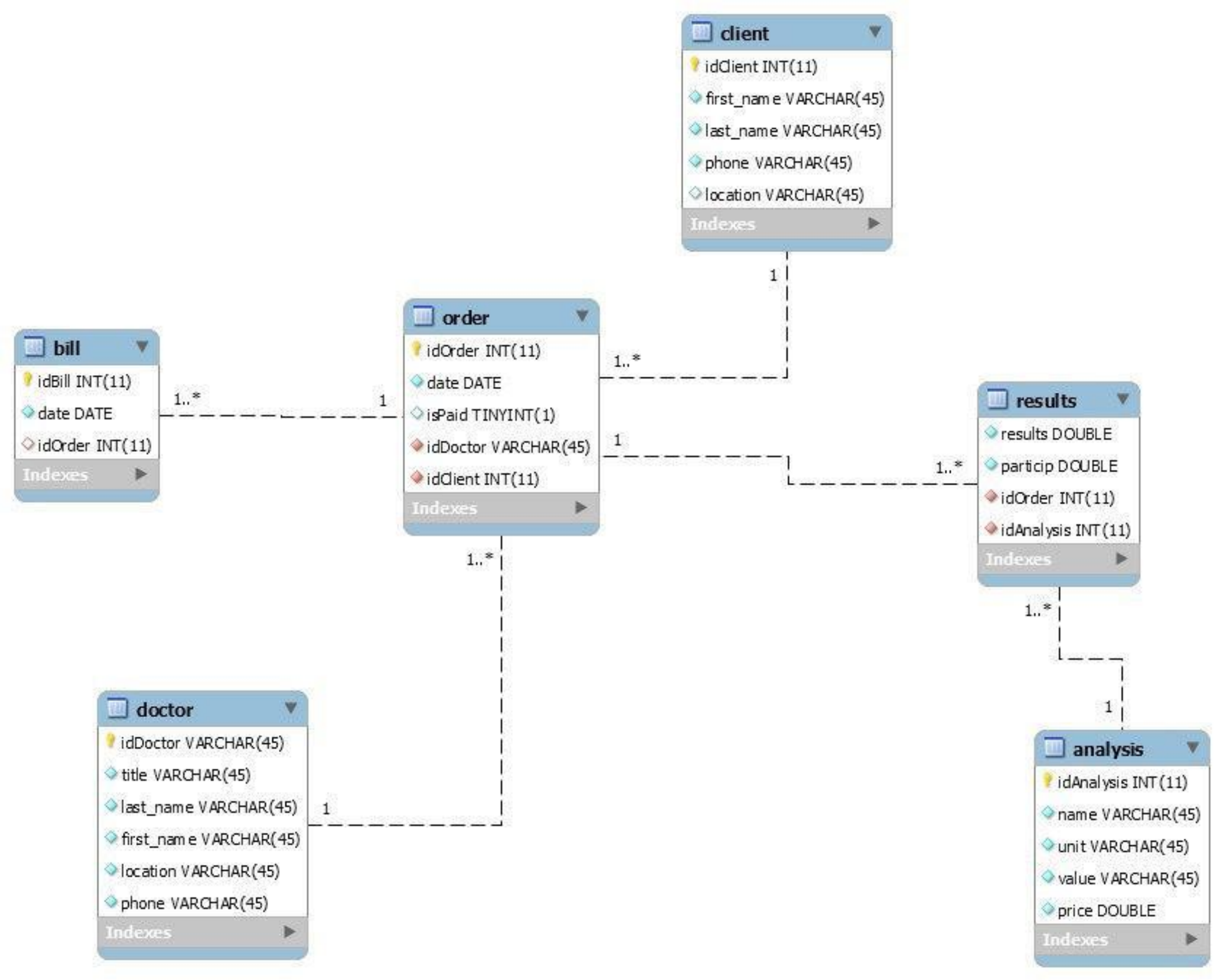

OBAL Software
1-Data Base
2-UML
3-Development
4-testing
1.1-Conceptual Data Model CDM
1.2-Logical data model
1.3-Script files for creating tables and inserting data
2.1-Use case diagram
2.2-Class diagram
2.3-Sequence diagram
3.1-add client
3.2-add analysis
3.3-search client
3.4-add results
3.5-print results and bill

Entity Association schema



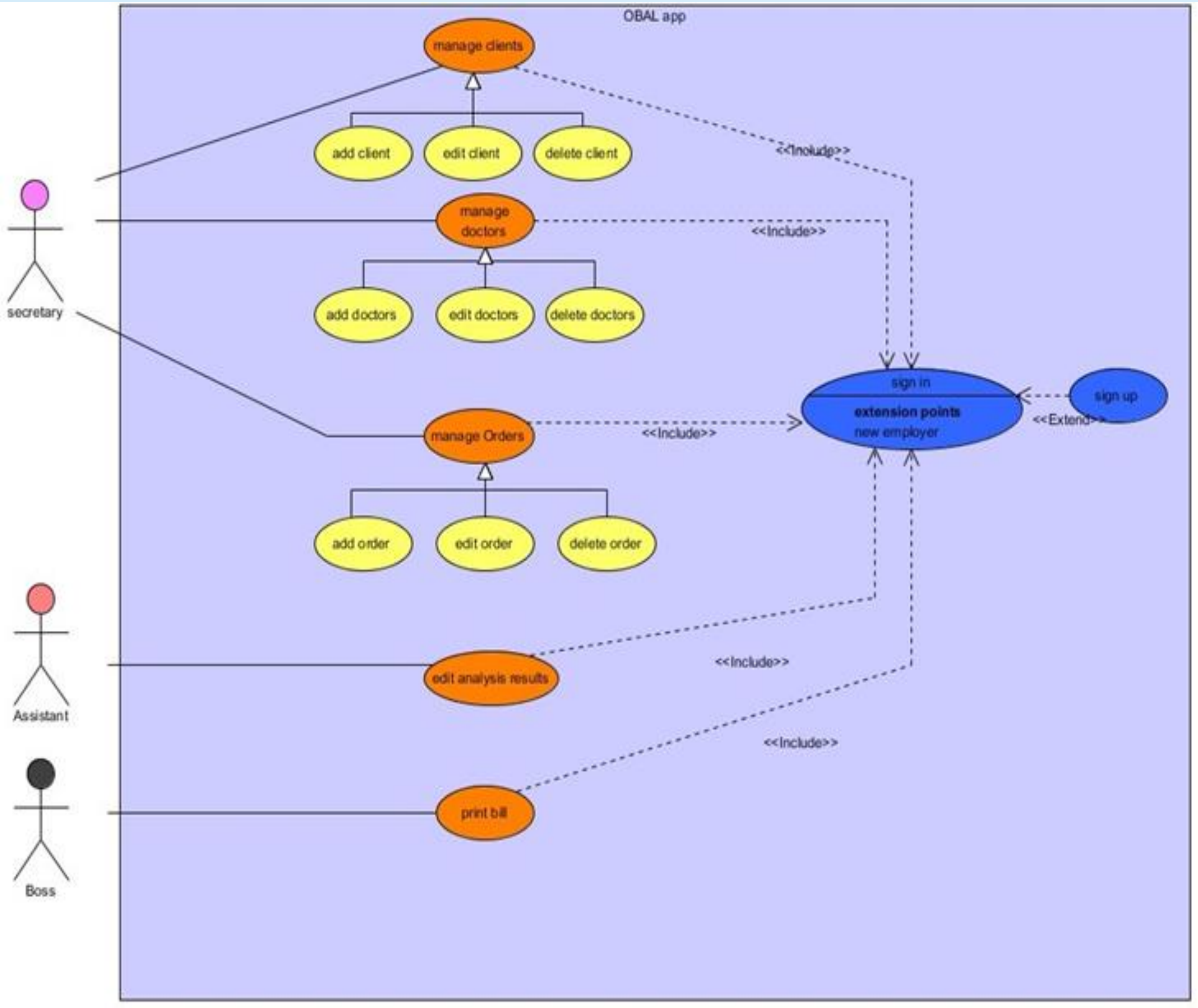
Database schema

- Client (idClient, first_name, last_name, phone, location)
- Order (idOrder, date, isPaid, idDoctor, idClient)
- Analysis (idAnalysis, name, unit, value, price)
- Doctor (idDoctor, title, first_name, last_name, location)
- Results (results, particip, idOrder, idAnalysis)
- Facture (idBill, date, idOrder)



Chapter 3: System design

Use Case diagram

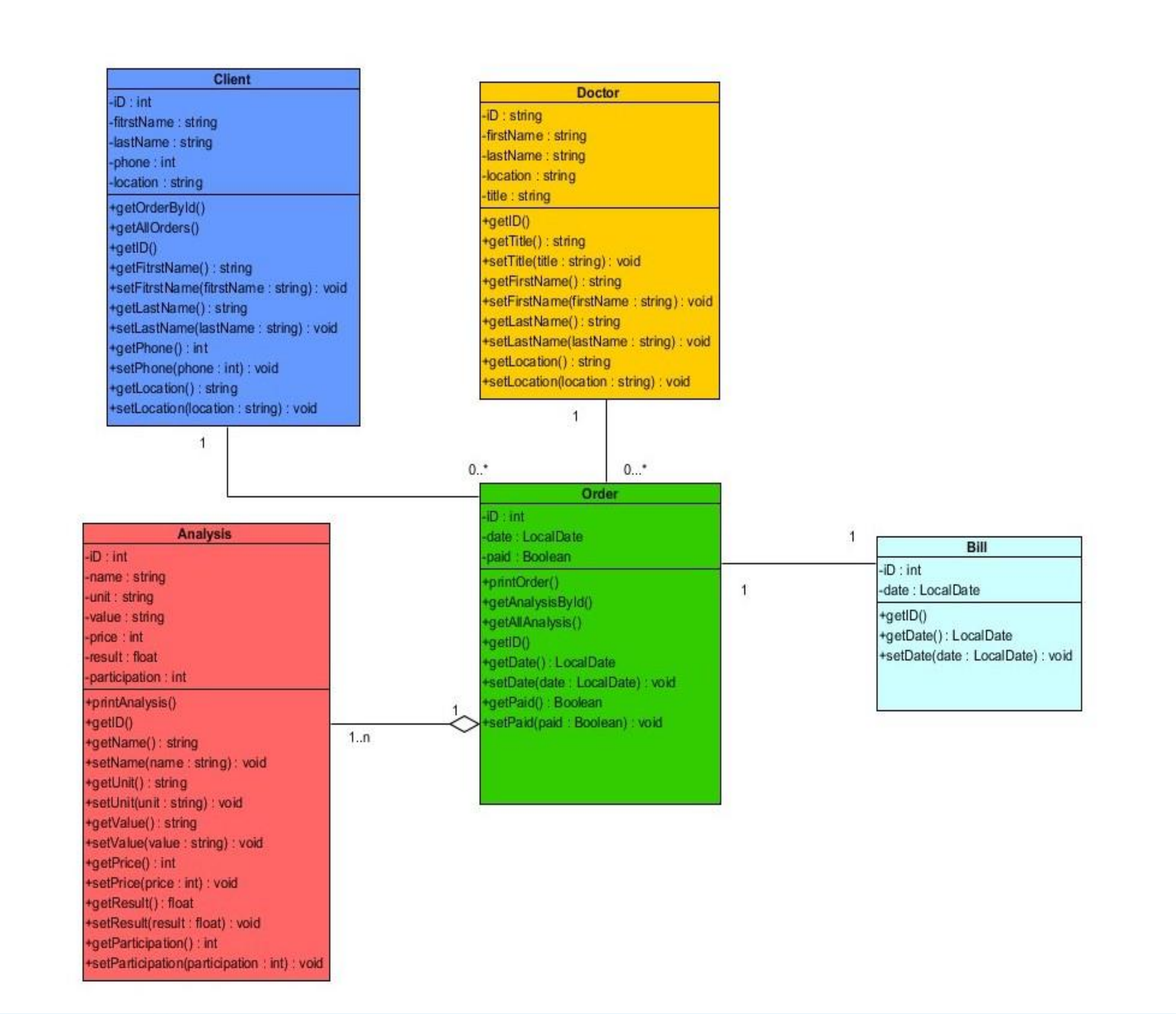


Textual description

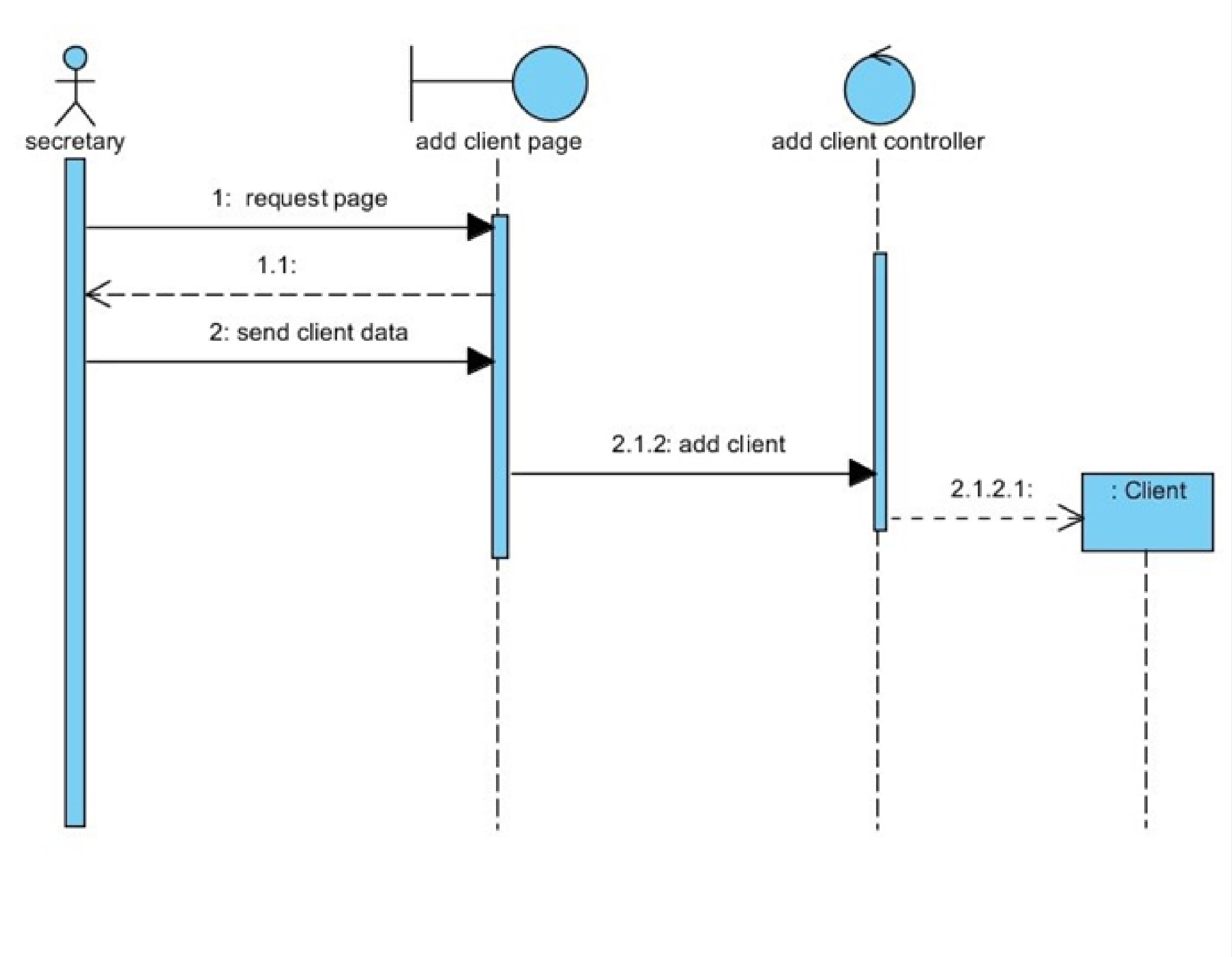
UC	Add Client	
ID	1	
Brief Description	Add a new Client	
First Actor	Employee	
Second Actor		
Preconditions		
Main Sequence	**User**	**System**
	1. Presses the ***CLIENTS*** button 3. Enters the client information (First Name, Last Name, phone, location) 5. clicks on ***SAVE*** button	2. Shows the ***ADD CLIENT*** page 4. If any information is missing, the system will warn the employee 6. Saves the information in the database tables
Post conditions	New client Added	
Alternate Sequence		

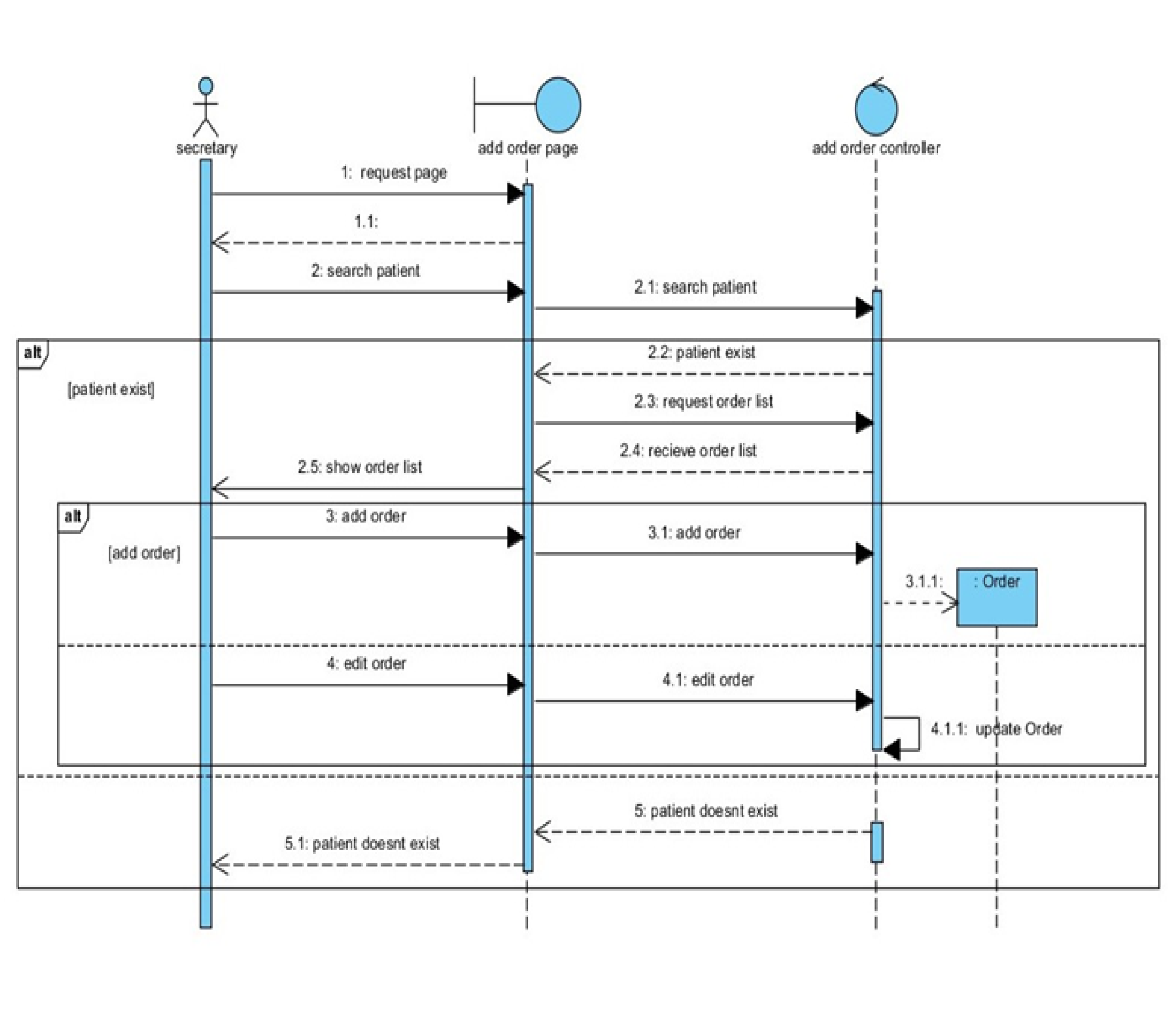
UC	Print bills	
ID	7	
Brief Description	Add new order for a client with list of analysis	
First Actor	Employee	
Second Actor		
Preconditions		
Main Sequence	**User**	**System**
	1 .Presses the ***BILLS & PAYMENTS*** button 3. Enters client's name and press search 3.1- Selects a client 3.2- Chooses an order 3.3- Chooses the analysis 3.4- Adds the participation value for each analysis 5. Clicks on ***SAVE*** button 7. Clicks on ***DONE*** button	2. Shows the ***BILLS*** page 4. Shows all the clients whose names are the same, or begin with the same letters 4.1- Shows the client's unpaid appointments 4.2- Shows order's analyses 4.3- Shows the participation's textbox 4.4- Saves the participation value 6. Saves the entered information 8. Opens ***BILLRESULTS*** page
Post conditions	Bill printed	
Alternate Sequence		

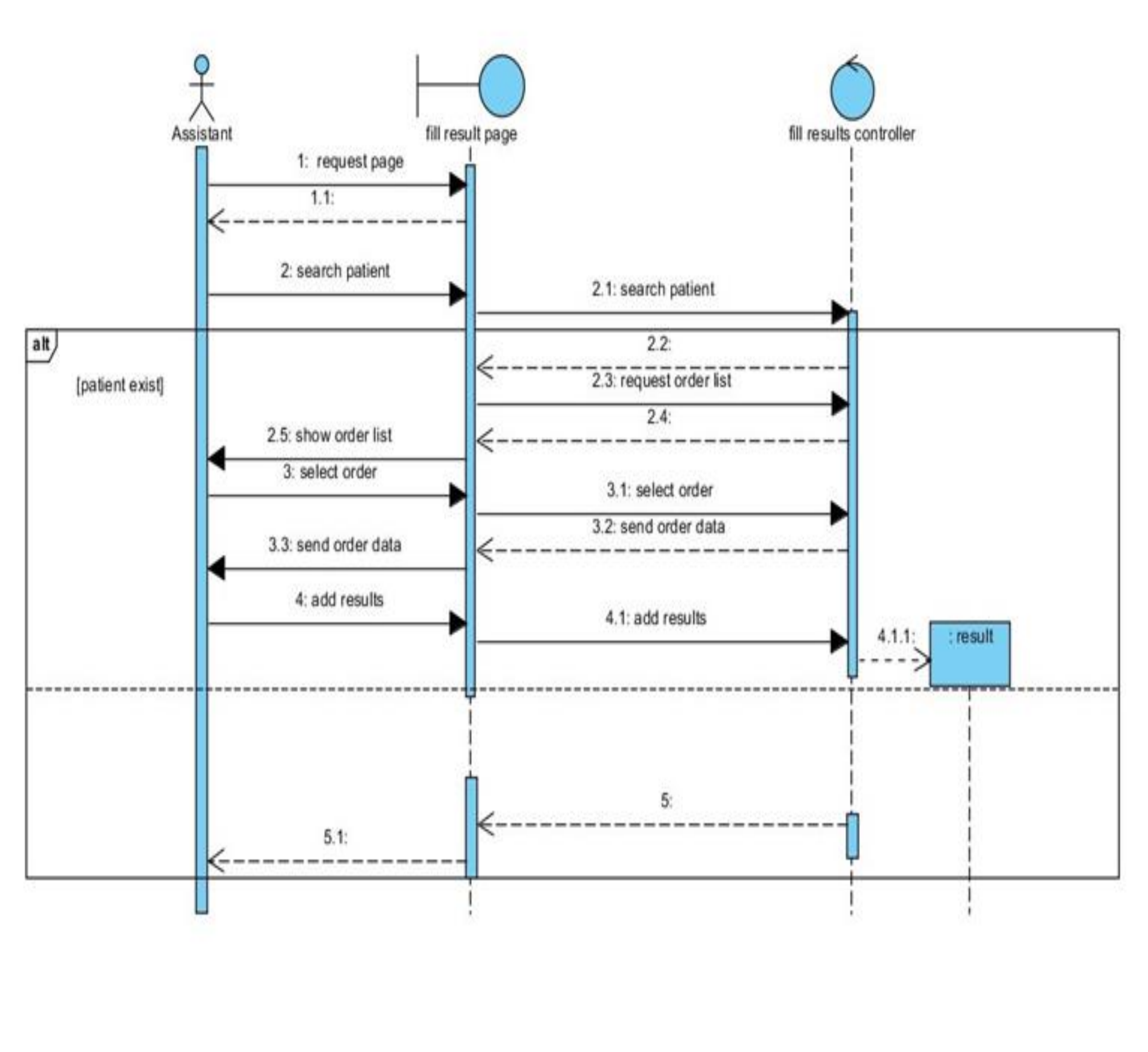
Class diagram

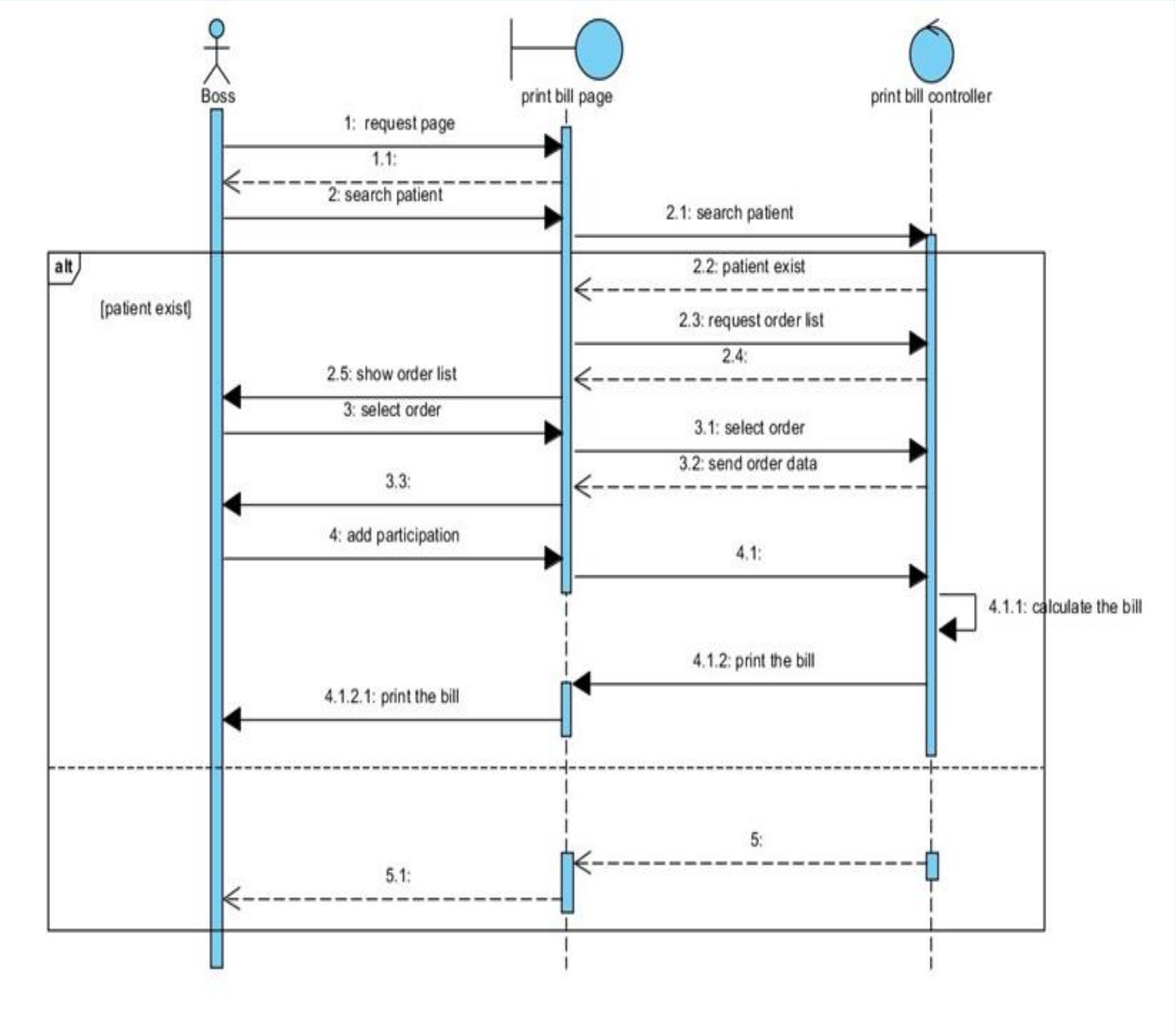


Sequence diagram

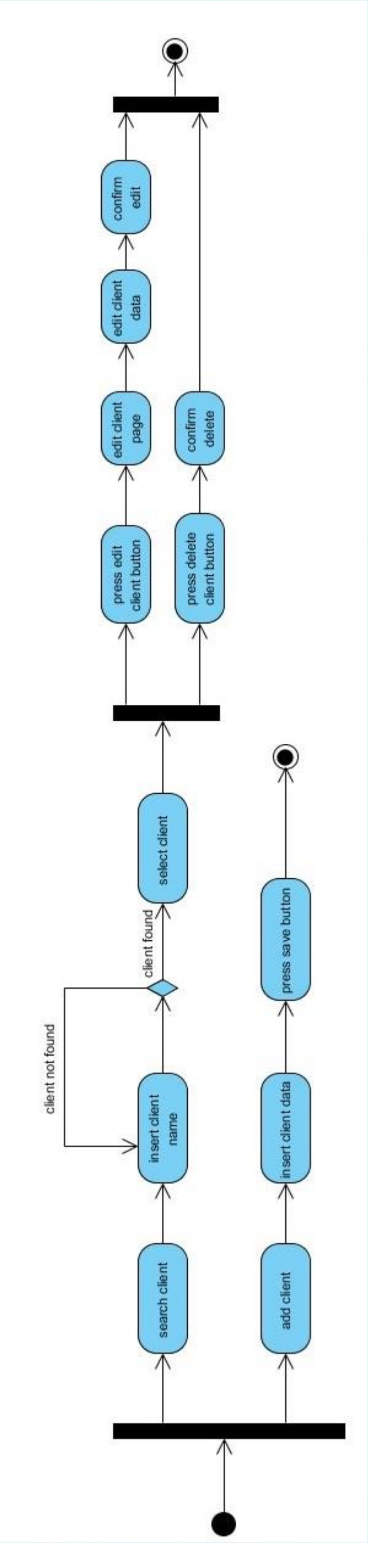



secretary
add order page
add order controller
1: request page
1.1:
2: search patient
2.1: search patient
alt
[patient exist]
2.2: patient exist
2.3: request order list
2.4: recieve order list
2.5: show order list
alt
[add order]
3: add order
3.1: add order
3.1.1:
: Order
4: edit order
4.1: edit order
4.1.1: update Order
5: patient doesnt exist
5.1: patient doesnt exist


Assistant
fill result page
fill results controller
1: request page
1.1:
2: search patient
2.1: search patient
alt
[patient exist]
2.2:
2.3: request order list
2.4:
2.5: show order list
3: select order
3.1: select order
3.2: send order data
3.3: send order data
4: add results
4.1: add results
4.1.1:
: result
5:
5.1:


Boss
print bill page
print bill controller
1: request page
1.1:
2: search patient
2.1: search patient
alt
[patient exist]
2.2: patient exist
2.3: request order list
2.4:
2.5: show order list
3: select order
3.1: select order
3.2: send order data
3.3:
4: add participation
4.1:
4.1.1: calculate the bill
4.1.2: print the bill
4.1.2.1: print the bill
5:
5.1:

Activity diagram



Conclusion

Throughout this project, we have been led to develop a desktop app. This app was an opportunity to apply the knowledge we've gained during the third year training.

However, we've faced some difficulties working on Github, but we figured them out.

Although, due to the lack of time, we weren't able to apply the "Multi user login" feature, so we ended up doing the app with no authentication.

It allowed us to implement the achievements of the design and modeling UML, JAVA, and SQL, and know the steps to follow in future projects.