**University of Bedfordshire**

**Assignment 1 Individual Assignment**

**Title of Project:**

**Control an Elevator C#**

**Submitted By:** Sushant Rajkarnikar (2026582)

**Table of Contents**

[1. Introduction 4](#_Toc119069412)

[2. Aims & Objectives 4](#_Toc119069413)

[3. Requirements 4](#_Toc119069414)

[Hardware Requirement 4](#_Toc119069415)

[Software Requirement 5](#_Toc119069416)

[Functional Requirement 5](#_Toc119069417)

[4. Diagram 5](#_Toc119069418)

[Use Case Diagram 5](#_Toc119069419)

[Activity Diagram 6](#_Toc119069420)

[5. Task Description 7](#_Toc119069421)

[Graphical User Interface(GUI) 7](#_Toc119069422)

[Event Handling 7](#_Toc119069423)

[Database design 9](#_Toc119069424)

[Delegation and Timer 10](#_Toc119069425)

[5.5 Optimization 11](#_Toc119069426)

[6. Testing 12](#_Toc119069427)

[7. Conclusion 13](#_Toc119069428)

**Table of Figures**

Figure 1: Use Case of Lift System 5

Figure 2: Activity diagram 6

Figure 3: GUI of Lift System 7

Figure 4: First floor call button Pressed 8

Figure 5: Ground floor button pressed 8

Figure 6: Database design 9

Figure 7: Data display from database in GUI 9

Figure 8: Data stored in MS Access database 10

Figure 9: Lift door open between time interval 10

Figure 10: Lift move upward 11

Figure 11: Try-catch block code 11

# Introduction

The design and system of an elevator has a vital role in enhancing accessibility to all floors and is more effective than climbing a staircase. Thanks to rapid development in modern lift technology and quick alterations, passengers can now traverse even hundreds of floors in under a couple of seconds. Controlling an Elevator, the first assignment, is has a weight worth 30% of the overall grade for the unit on Object-Oriented Programming and Software Engineering. Through this assignment, the student's knowledge of the OOPs concept and the C# (CSharp) programming language was tested. The student had to create a working elevator system with its control for simple interaction and a database for data storage for this project. It was also required to run and do the testing of the application and come up with a system design during the assignment. The project will be successful if all of the university's given requirements were met.

# Aims & Objectives

This assignment's primary goals were to examine the intricate challenge involving the lift mechanism and to build problem-solving abilities. The assignment's goals are listed below:

1. To create a visually appealing GUI design with buttons and a panel board for controlling the lift.
2. Thorough analysis of the OOPs and C# programming concepts
3. To build a functional database to store the data.

# Requirements

## Hardware Requirement

1. At least 4 Gb of RAM is required but 16 Gb is preferred for better performance.
2. A 64-bit processor running at 1.8 GHz or faster; quad-core or better is advised. The ARM architecture is not supported.
3. Depending on the installed features, 850 MB to 210 GB of free space is required; normal installations calls for 20–50 GB of free space.

## Software Requirement

1. Visual studio 2022
2. Visual Studio requires the .NET Framework 4.5.2 or later for installation. The .NET Framework 4.8 is required for Visual Studio to function.
3. Windows 10 64 bit or higher OS
4. Visual paradigm
5. Microsoft Access

## Functional Requirement

1. A call button to shift the elevator shaft to the desired floor.
2. A button that shows the elevator movement details log from the database.
3. Lift doors that open and close in response to floor buttons being pressed.
4. A panel board with buttons and floors number display to control the lift.

# Diagram

## Use Case Diagram

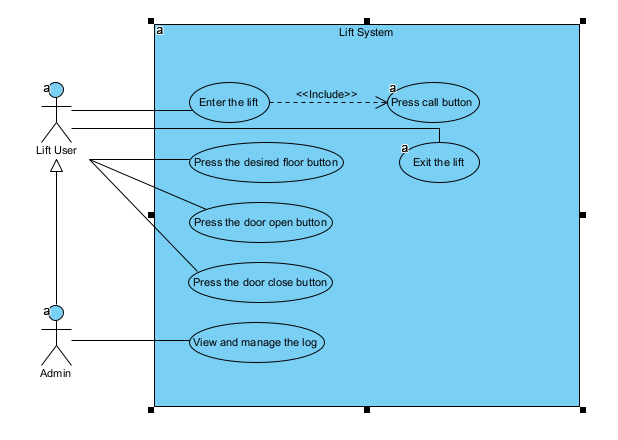


Figure : Use Case of Lift System

The Lift user must press the call button for the lift door to open and then he/she can enter it as shown in the above use case diagram. The user clicks the desired floor's button after stepping into the lift. The user exits the lift when it has reached the desired floor and the door automatically opens. The user can also open or close the lift doors manually by clicking the respective buttons when inside the lift. The administrator can similarly review the activity log data that is stored in the database.

## Activity Diagram

## 

Figure : Activity diagram

First the user goes up to the lift then clicks the call button for the lift to arrive at the called floor. The mechanism then initially starts as shown in the diagram above. The system then verifies the floor from where the request originated when the user presses the call button. The lift door opens automatically when it reaches the called floor. Then the person/s can enter the lift and click on the desired floor button he/she wants to go on. If the user clicked the first floor button while the lift is in ground floor it will move to the first floor and vice versa. Additionally, the door opens after the lift has reached the appropriate floor. The mechanism ends and after the person has exited the lift the door closes after some time. If the door closes before the person can exit the lift, he or she must press the open button to unlock the door. When the user exits the elevator after reaching the floor, the door automatically closes after a brief period of time, signaling the conclusion of the system.

# Task Description

## Graphical User Interface(GUI)

## 

Figure : GUI of Lift System

This is the GUI of the entire lift control system interface. The floor buttons, open or close door choice buttons, and a screen that shows the current floor number are all located on a control panel board. In addition, there is a show button that shows the history of lift operations.

## Event Handling

When the user presses any of the buttons, such as call, floor, open, shut, or show log, a specific job is carried out in the event handler.

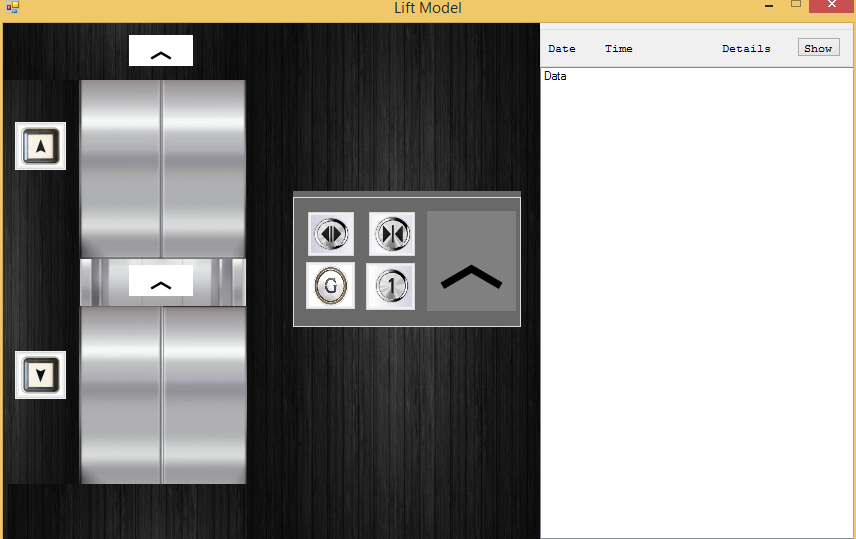


Figure : First floor call button Pressed

When the first floor button is pressed in the image above, the lift shaft ascends to the relevant floor.

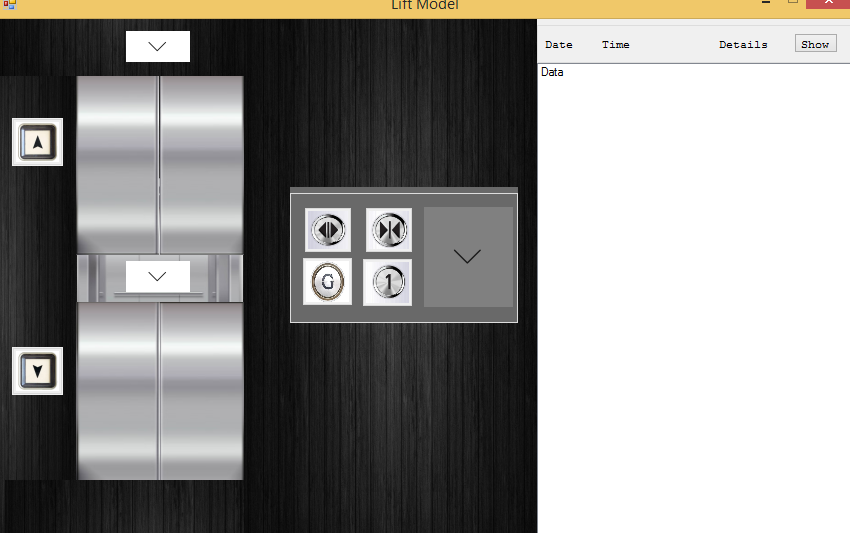


Figure : Ground floor button pressed

The elevator shaft in the above illustration went to the ground floor when the ground floor request button was touched, indicating that the button handled an event.

## Database design

## 

Figure : Database design

With the aid of Microsoft Access, a database was built in order to hold the corresponding details, date and time information for the lift. The table's name is LiftTable and it has the field names SN, Date&Time, and Details. The database is called LiftDb.

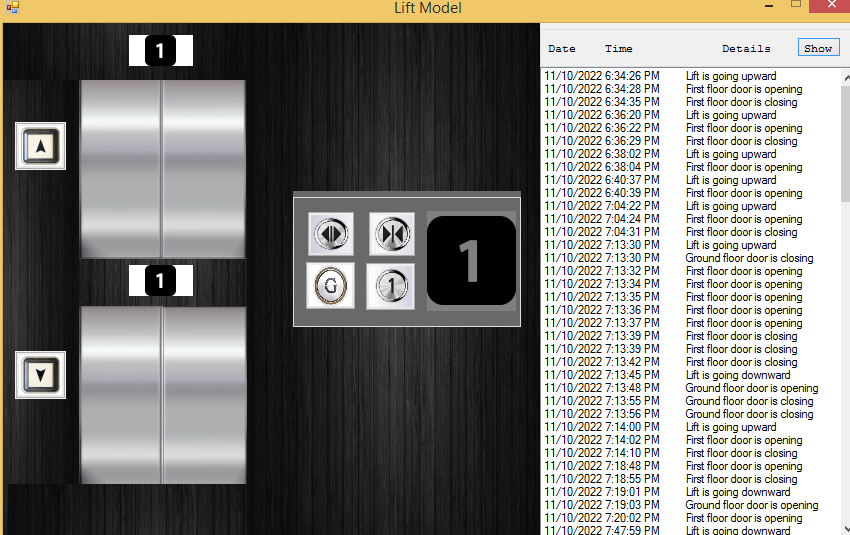


Figure : Data display from database in GUI

The logbook at the left that was intended to record the lift's details together with the date and time is displayed in the above illustration.

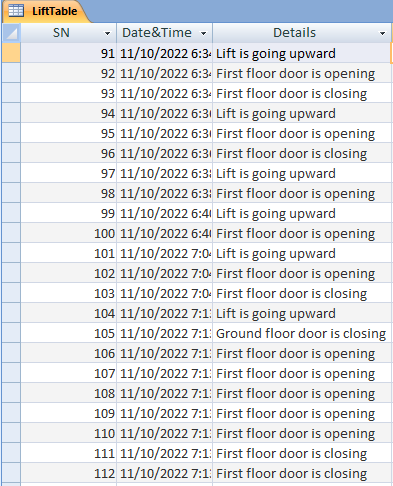


Figure : Data stored in MS Access database

The lift operation record history was successfully stored in the relevant database, as seen in the above figure. Additionally, it displayed thorough information along with the precise time and date.

## Delegation and Timer

## 

Figure : Lift door open between time interval

The timer has also been used to open or close the door in a brief amount of time. I can implement those event handling and callback functions with the aid of delegates.

## 

Figure : Lift move upward

In the above mentioned figure the task was completed to ensure that each event handler was successfully executed. The elevator shaft glides up and down smoothly thanks to the "Timer" in the buttons.

## 5.5 Optimization

Use a relative path rather than an absolute path to execute the application efficiently on any window framework because relative paths do not contain the domain name and links will continue to function without needing to be changed. The assignment folder's resources folder contain the image icons that were used in the assignment. The bin folder contains the database file which was built inside of it.

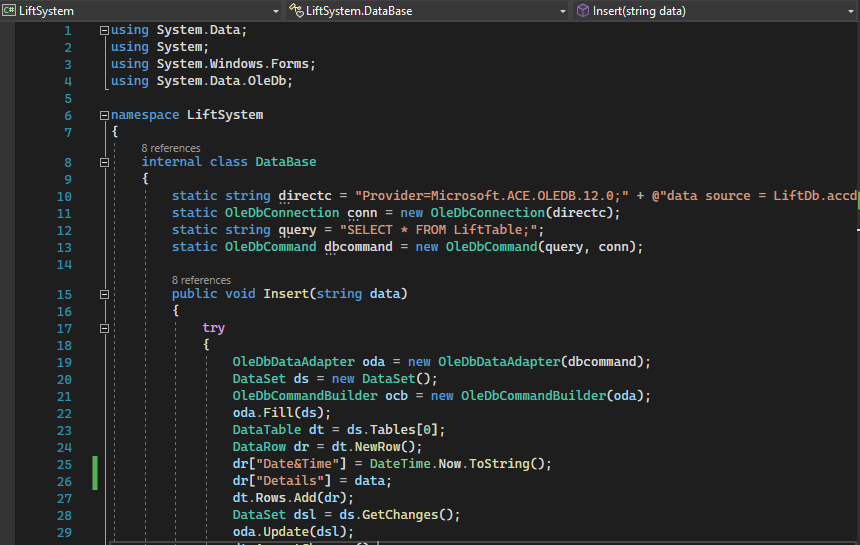


Figure : Try-catch block code

The try-catch block was used to manage failures and keep the database-related functions from crashing in the above diagram.

First, all logical faults in the system were found and fixed in order to maximize its stability. The try-catch block was also widely utilized in those areas of the code where it was required.

# Testing

The last step of the project is testing, which is done to identify any faults or errors so they may be corrected before being released.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.N** | **Task** | **Intended Outcome** | **Real Outcome** | **Comments** |
| 1. | Press the call button | When the lift reaches the appropriate floor, the door automatically opens. | Same as anticipated outcome. | Fine |
| 2. | Press the First or Ground floor button | The lift moves to the first or ground floor as per the button clicked and door open after few seconds automatically. | Functioning well | Fine |
| 3. | Floor numbers are visible on the control panel board. | As the elevator moves to the next or previous floor, the floor number changes. | Functioning well | Fine |
| 4. | Activity history of lift | Showed the state details of the lift along with the date and time in the log. | Same as anticipated outcome. | Fine. |

# Conclusion

OOPs and the C# programming language were used to finally finish the assignment. It took a lot of time and was chaotic, but with persistence and hard effort, it was finished on schedule. I learned a lot from this assignment to be able to tackle challenges this complex in the real world in the near future. The Teachers helped and supported me greatly throughout the project by giving me their sound advice that I needed to finish it.