



# REPORT

APPOINTMENT SYSTEM USING BST AND HASHTABLE

SHUBHAM JAIN

A00258743

## Abstract

The objective of this project was to develop an Online Appointment System in Java. The purpose of implementing this application was to create a system through which a person can easily choose and make an online appointment for any purpose just by sitting at home.

Online appointment system aims to improve appointment scheduling by bringing all manual appointment process of the city at one platform, eliminating long waiting lines.

The application was successfully implemented by using Java programming languages. This application does not aim to target any specific group but every individual who wants to seek help managing appointments and that is why it was kept in mind to keep the user interface simple and friendly while building this application.

## Introduction

Booking appointment online has become a new trend in the past few years and is considered as one of the key processes in the industry. Bailey (1952) considered scheduling system as a trade-off or a compromise between a company and client's waiting times. Customers who get late for the appointments or who fails to come becomes the reason for the underutilization of a company's time. Idle time and underutilization of company's time are also resulted by gaps in the appointment times.

The aim of this project is to create a platform where companies can access/interact efficiently with each other and provide ease and comfort to the clients. It also aims to resolve the problems that companies have to face while taking appointments and keeping files.

## Scope

The scope of the project is limited to the company side. No client can access the platform to generate appointments. Only admin has the right to login and create a unique reference number for an appointment. He can give the client that reference number and can query appointment data with reference number.

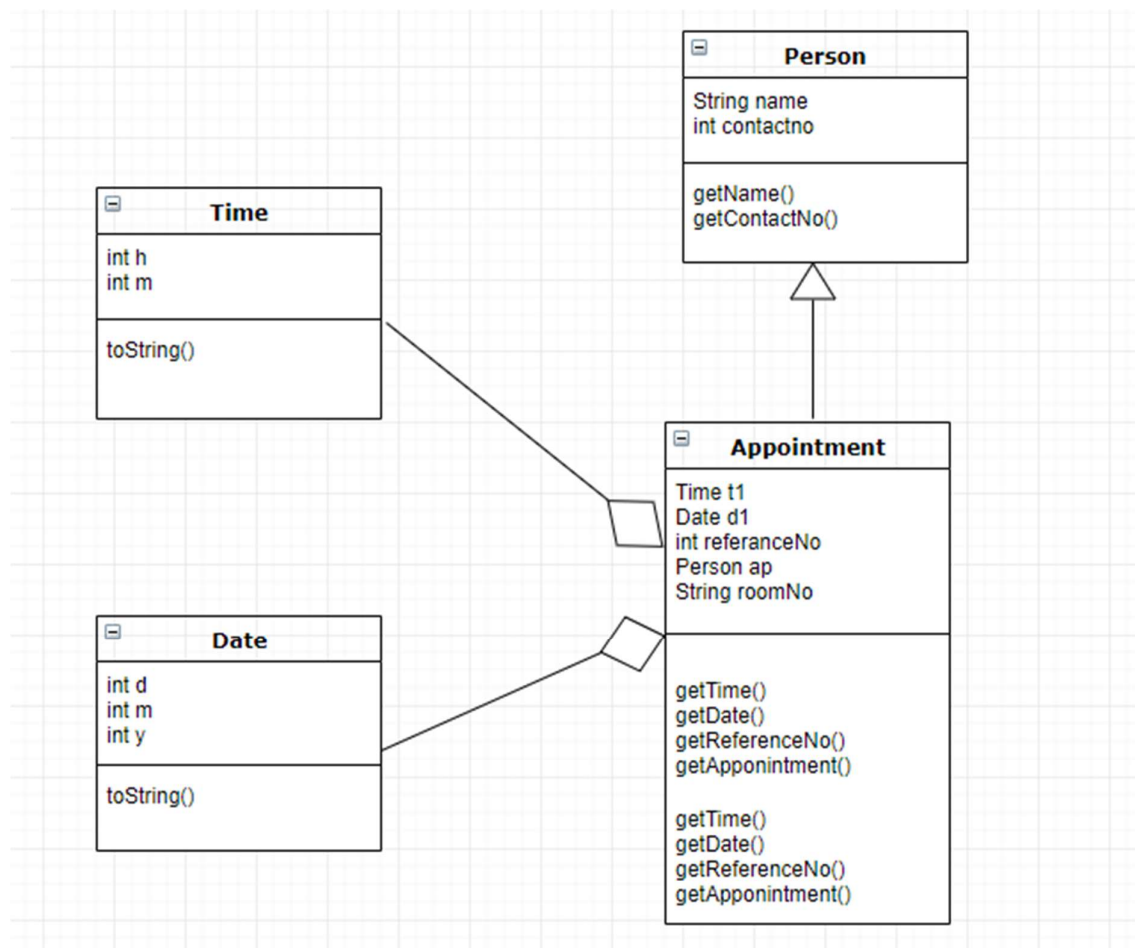
## Limitations and Future Implementation

The biggest limitation being time. This application remains only on server side, database can be implemented and client can be provided an interface to book their own appointment with a person within available time slots.

## Features

- USE OF BINARY SEARCH TREE AND HASH-TABLE DATA STRUCTURES TO INSERT AND SEARCH DATA.
- ITERATOR TO VIEW DATA
- USED SINGLETON PATTERN TO PASS SAME INSTANCE OF BST AND HASHTABLE.
- MVC PATTERN IMPLEMENTED TO MAKE CODE MORE UNDERSTANDABLE.
- AGGREGATION OF TIME, DATE AND PERSON CLASS.
- INHERITANCE OF PERSON AND APPOINTMENT CLASS.
- REFERENCE NUMBER GETS AUTOMATICALLY GENERATED WITH 4 NUMBERS FOLLOWED BY 4 CHARACTERS.
- JTABLE FOR REPRESENTATION OF DATA ON ANOTHER FRAME, USE OF STATIC VARIABLES TO RETAIN INFORMATION.
- DATA PASSING BETWEEN EACH FRAMES ON BUTTON CLICKS SO THAT THE SAME LIST IS ACCESSED IN ALL FRAMES.
- EFFICIENT MEMORY MANAGEMENT TO RELEASE RESOURCES WHEN REQUIRED EG. DISPOSING THE FRAME IN THE BACKGROUND.


## UML



- Created 2 Appointment Classes- BSTAppointment and HashTableAppointment
- BSTree and Hashtable are classes for data structures being used.

## Screenshots

Main Frame



The screenshot shows the main frame of the Appointment Management Software. It features a title bar with a small icon and standard window controls. The main content area has a light gray background with the title "Appointment Management Software" in a large, bold, black font. Below the title, there are two columns of controls. The left column is titled "Search Appointment" and contains a text input field labeled "Enter Reference No." and a "Find" button. The right column is titled "Create Appointment" and contains an empty text input field and a "Create" button. At the bottom center, there is a "View All" button.

# Appointment Management Software

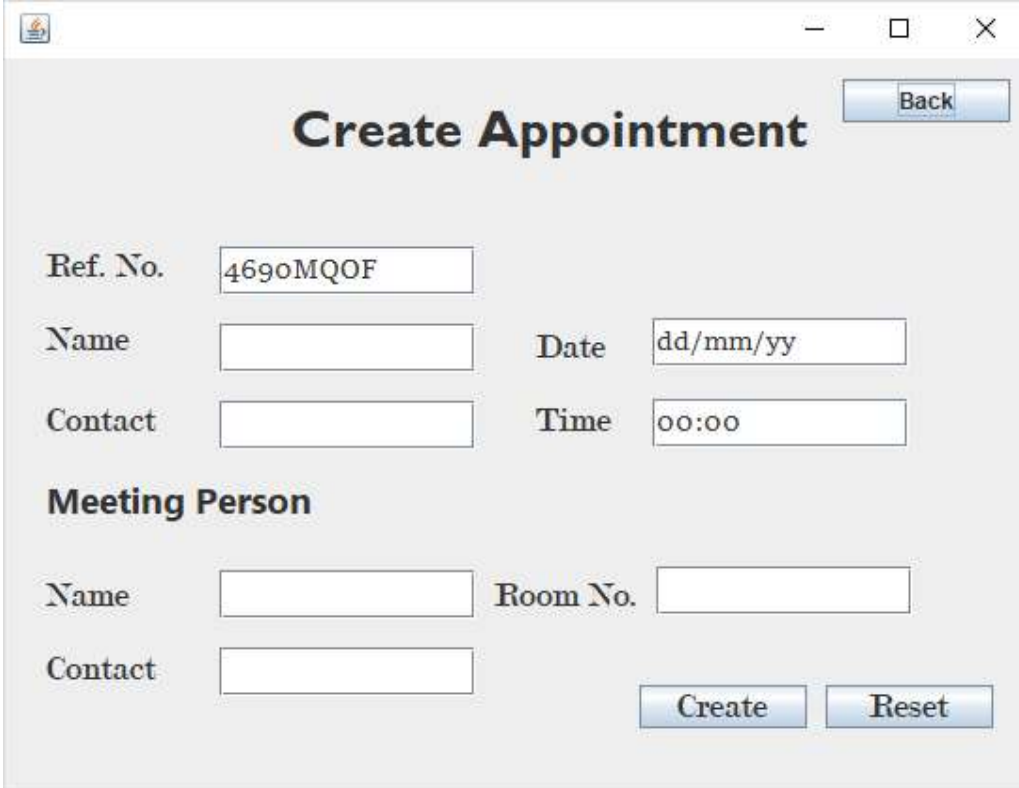
Search Appointment      Create Appointment

Enter Reference No.     

Find      Create

View All

Create Onclick()



The screenshot shows the "Create Appointment" form. It has a title bar with a small icon and standard window controls. The main content area has a light gray background with the title "Create Appointment" in a large, bold, black font. In the top right corner, there is a "Back" button. The form contains several input fields: "Ref. No." with the value "469oMQOF", "Name", "Date" with the placeholder "dd/mm/yy", "Contact", and "Time" with the placeholder "00:00". Below these fields, there is a section titled "Meeting Person" with "Name", "Room No.", and "Contact" input fields. At the bottom right, there are "Create" and "Reset" buttons.

## Create Appointment

Back

Ref. No. 469oMQOF

Name      Date dd/mm/yy

Contact      Time 00:00

### Meeting Person

Name      Room No.     

Contact     

Create      Reset

Find Onclick(Input Reference)- Uses Binary Search Tree

Delete

Back

Ref #No.	Name	Contact	Meeting Name	Meeting Contact	Date	Time	Room
1234ABCD	Ericsson	9876543210	Shubham	8855082050	02/11/18	09:15	V205

ViewALL onlick -Uses Hashtable

Delete

Back

Ref #No.	Name	Contact	Meeting Name	Meeting Contact	Date	Time	Room
3434sdff	Ericsson	9876543210	Shubham	8855082050	02/11/18	09:15	V205
5575dfgg	Ericsson	9876543210	Shubham	8855082050	02/11/18	09:15	V205
1245gfh	Ericsson	9876543210	Shubham	8855082050	02/11/18	09:15	V205
1234ABCD	Ericsson	9876543210	Shubham	8855082050	02/11/18	09:15	V205

## Performance

Cost of search and insert BST.

### Theorem.

Here keys are inserted in random order, then height of tree is  $(\log N)$ , except with exponentially small probability. Thus, search and insert take  $O(\log N)$  time.

### Problem.

Worst-case search and insert are proportional to  $N$ . If nodes in order, tree degenerates to linked list.

Cost of search and insert BST.

### Theorem.

Ideally all the time complexities should be  $O(1)$  according to the code.

### Problem.

But  $O(1)$  is achieved only when number of entries is less than number of buckets. In other words if load-factor is less than 1.

Worst Case is always  $O(n)$ , You can go about looking-up all the elements in the list

## Conclusion

A- simple appointment management system is created by using Binary Search Tree and Hash table as data structures, values are iterated using Binary Search Tree and different design patterns are applied. Also object oriented concept are kept in mind as seen in the features.