**Dental Appointment System**

1. **Features of Dental Appointment System**

In this dental appointment system, there are total of 5 features, 3 features for patient and 2 features for admin usage.

**For Patient**

1. **Make a dental appointment** – patients can make an appointment using this system by choosing the date and time session as well as the dentist that they want for a particular appointment. After selecting the time and date, a search function feature is also incorporate to display the available dentist for the chosen date and time. After making the appointment, the appointment information will be printed out with the appointment id along with the time and date as well as the dentist’s name.
2. **List available times of selected dentist** – this feature allows the patient to see the available time slots of a chosen dentist on a chosen date.
3. **Edit a dental appointment** – patients can edit an appointment by providing the appointment id that was given to them previously. A confirmation message, including the appointment details, will be asked to see if the patients still want to edit his/her appointment. If patients choose to edit his/her appointment, a new time and date will be asked along with the dentist that they want for their edited appointment. Likewise, a search function feature is also incorporate to display the available dentist for the new chosen date and time. Finally, the appointment information will be printed out with the appointment id along with the new time and date as well as the dentist’s name.

**For Admin**

1. **Browse appointments for a dentist** – this feature allows the admin to browse all the appointment for a chosen dentist by entering the dentist’s name
2. **Search for appointment** – this feature allows the admin to search for a particular appointment by entering the appointment id.
3. **Data Structures and Format**

**Data type (refer to the powerpoint slide for pictorial representation):**

1. **Datetime.go** – the data structure used for the accessing the month is an array of pointer of size 12 (each representing a month from Jan to Dec) so it only takes constant time complexity, O(1) to access each month. Each pointer from the month will consist of another array of pointer of size 31 (each representing a day in a month) which also takes constant time complexity, O(1) to access it. 31 was used as the maximum number of days in a month is 31. Each pointer from the day will point to the address of binary search tree (BST) which will consist of a total of 15 time slots from 9am to 5pm (each slot is 30minutes) represented by 1 to 15. The BST is initialized such that it is a balance tree (*refer to the diagram in powerpoint slide*) so that adding or removing a time slot will have a worst time complexity of O(log2N), where N is the number of timeslots. However, the BST can be further improved into Adelson-Velsky-Landis (AVL) tree which is a self-balancing tree.
2. **Dentist.go - t**he data structure used for accessing/storing/searching the appointment is an array of pointer of size 10. The dentist’s name will be passed to the hash function to compute the index to store the information at that particular position. Information like dentist’s name, dr. id as well as the available time slot will be stored in this data structure. The available time slot will take on the data structure of the datetime which consists of the month, date arrays as well as the BST time slot.

The index will be computed using the hash function:

1. By dentist name - by adding the ASCII number of dentist name and then get the remainder when divided by size of array which is 10 so each dentist and his/her information will be stored at the same index for the array. Likewise, A linked-list is used for collision of the same index to store appointment with the same index.

In this way, the time complexity of adding/modifying/searching timeslot of dentists will be between O(log2N), and O(N), where N is the number of dentists.

1. **Appointment**.go – the data structure used for accessing/storing/searching the appointment is an array of pointer of size 10. There will be 2 such arrays, one for storing the appointment by passing the appointment id to the hash function while another is for storing individual dentist appointment by passing the dentist names to the hash function.

The index will be computed using the hash function:

1. By appointment id – by getting the remainder when divided the appointment id by size of array which is 10 so each appointment will be store on the array before the index is repeated. A linked-list is used for collision of the same index to store appointment with the same index.
2. By doctor id - same as above except this time is by doctor id

In this way, the time complexity of adding/modifying/searching an appointment will be between O(1) and O(N), where N is the number of appointments.

1. **Core Go Files**
2. Main.go – this is the entry point for the application and will check for whether the user is an admin or a patient, after which will direct the user to the specific direction
3. Patient.go – this file is one of the core files that have all the functions for patients to make/edit appointment as well as searching for available times for a particular dentist
4. Admin.go – this file is one of the core files that have all the functions for the admins to search a particular appointment by keying in the appointment id or to browse all appointments for a particular dentist
5. Utils.go – this file consists of some utility functions like clear screen, convert month from string to int, checking if a year is a leap, convert to Title case etc.
6. **Error Handling and Concurrency Mechanism**

All the datatype function method will return error message if there is any error. A ‘defer’ anonymous function for panic recovery is also declared at beginning of the functions in admin.go and patient.go where the core part of the application are.

Goroutines were used when the system updates the data structure for the appointment and dentist list as these will not affect the core function of the application. Concurrency issue is prevented by only allowing the system to remove/update the timeslot when making/editing appointment by specify a buffered channel with size of 1.

1. **Instructions to run the application**
2. Change the directory to the folder that consists of all the go files
3. Key in ‘go run .’ to run the application
4. The application will ask for whether you are ‘admin’ or ‘patient. Please key in the correct choice. If wrong choice is keyed, you will be asked whether to continue with the application again.
5. When keying the month, please key in the spelling of the particular month rather than the integer.
6. After you have completed making/editing an appointment, a message will be asked to see if you still want to continue making appointment. Key in ‘Y’ if you want to continue else ‘N’ if you want to exit from the application.