Book My Journey

5

Generated by Doxygen 1.8.13

Contents

Chapter 1

README

BOOK MY JOURNEY

This projects creates an application which helps the user to book a ticket for the journey.The main part of the project is to maintain concurrency when multiple users request for booking the same seat.

Getting started:

Prerequisites

You need to have a git environment in your local machine.
Enter these commands

sudo apt install git

You need to have QT work frame to run the project.Download it from this link:

https://www.qt.io

After install run these commands:

sudo apt-get install libqt4-dev

Make sure you have proper version of it

Cloning and Developing

A quick introduction of the minimal setup you need to get a
copy of the project and running.Run the

git init
git clone https://github.com/IITH-SBJoshi/concurrency-5.git

The repository will be downloaded

Run BookMyJourney.pro using qt.

Build

Go to the directory where the project is extreacter and run this
command to build the project.

qmake -project && qmake && make

We can run it in qt also after opening the BookMyJourney.pro
A run buttun is present in the left bottom of the window or else
CTRL+R will also run the project

To run the project we should first establish a connection to server,for that make sure that the server is running.To run the server in your machine run server.cpp Run these commands :

g++ server.cpp -pthread -o output
./output

Testing and Features:

This application is a sample of ticket booking system with core
concept of concurrency when multiple users try to book same ticket
at the same time.When we run the project first dialog requires us toi
enter the account details to Log into the app.If the user doesn't
have an account he can create it also.

After log in ,user needs to select the origin and destination of

his/her journey and the date of the journey.Booking is allowed only
upto same date of next month.

After that he/she needs to select the seats he/she like.A disabled
seats means that it already booked and seats selected will be
displayed in red color.

After that user needs to confirm the terms and conditions and make
payment.A timer will be started and he/she should should make
payment before the timer ends or else the tickets will be lost.After
selecting the confirm booking that seats will be locked and before
the timer ends these seats cannot be booked by anyone.

When the server goes off accidentally the data is in the server backed up to into two files
named 1.details.txt -> which stores the details of user and his journey 2.seats.txt -> which stores the seatmatrices.

Testing

File test.cpp runs the test and makes sure that concurrency in the server is working fine.
It creates multiple users who try to book same ticket and ensures that only one user books it.
g++ test.cpp -pthread -o output
./output

Coding style tests:

These tests test where there are mistakes in
coding style in our project.Coding style used in this project is Google coding style.
To run the following test run this command
python cpplint.py <c++filename>

License:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

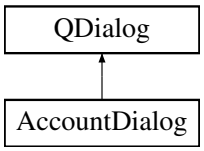
Class Documentation

4.1 AccountDialog Class Reference

The AccountDialog class.

`#include <accountDialog.h>`

Inheritance diagram for AccountDialog:



Public Member Functions

- AccountDialog (QWidget *parent=nullptr)

  AccountDialog - This constructor displays the User interface, which is for new users. It asks the new user to enter details to create an account. All these details are noted down in server temporarily and will be stored into a text file which acts as data-base. This user can now login using his/her username and password in future.

4.1.1 Detailed Description

The AccountDialog class.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 AccountDialog()

```
AccountDialog::AccountDialog (
            QWidget * parent = nullptr )  [explicit]
```

AccountDialog - This constructor displays the User interface, which is for new users. It asks the new user to enter details to create an account. All these details are noted down in server temporarily and will be stored into a text file which acts as data-base. This user can now login using his/her username and password in future.

Parameters

parent	- variable (of type QWidget $*$) which represents the Dialog.

The documentation for this class was generated from the following files:

- BookMyJourney/Headers/accountDialog.h
- BookMyJourney/Sources/accountDialog.cpp

4.2 accounts Struct Reference

The accounts class.

Public Attributes

- char username [100]

  username - Stores the username of USER. Each User have a unique username, we don't accept username which is already taken by another user.
- char name [300]

  name - Stores the name of USER.
- char contact [12]

  contact - Stores the contact of USER.
- char mail [100]

  mail - Stores the mail of USER.
- char password [100]

  password - Stores the password of USER.
- vector< vector< string > > v

  v - Stores the Journey Details of USER.
- vector< vector< int > > seat

  seat - Stores the seats booked by the USER.

4.2.1 Detailed Description

The accounts class.

The documentation for this struct was generated from the following file:

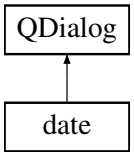
- server.cpp

4.3 date Class Reference

The date class.

`#include <date.h>`

Inheritance diagram for date:



Public Member Functions

- **date** (QWidget *parent=0)
- QDate selectedDate () const

  selectedDate - function that notes down the selected date.

4.3.1 Detailed Description

The date class.

4.3.2 Member Function Documentation

4.3.2.1 selectedDate()

```
QDate date::selectedDate ( ) const
```

selectedDate - function that notes down the selected date.

Returns

The documentation for this class was generated from the following files:

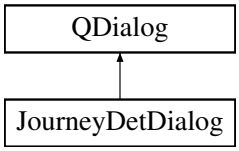
- BookMyJourney/Headers/date.h
- BookMyJourney/Sources/date.cpp

4.4 JourneyDetDialog Class Reference

The JourneyDetDialog class.

```
#include <journeyDetDialog.h>
```

Inheritance diagram for JourneyDetDialog:



Public Member Functions

- JourneyDetDialog (QWidget *parent=nullptr)

  JourneyDetDialog - This constructor displays the User Interface, which asks the user to enter the journey details. We can access to trains, which run in next 31 days. This is handled in date.h.

4.4.1 Detailed Description

The JourneyDetDialog class.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 JourneyDetDialog()

```
JourneyDetDialog::JourneyDetDialog (
            QWidget * parent = nullptr )  [explicit]
```

JourneyDetDialog - This constructor displays the User Interface, which asks the user to enter the journey details. We can access to trains, which run in next 31 days. This is handled in date.h.

Parameters

parent	- variable (of type QWidget ∗) which represents the Dialog.

The documentation for this class was generated from the following files:

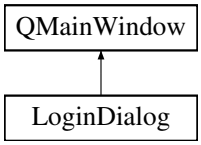
- BookMyJourney/Headers/journeyDetDialog.h
- BookMyJourney/Sources/journeyDetDialog.cpp

4.5 LoginDialog Class Reference

The LoginDialog class.

```
#include <loginDialog.h>
```

Inheritance diagram for LoginDialog:



Public Member Functions

- LoginDialog (QWidget ∗parent=nullptr)

  LoginDialog - This constructor displays the home-page, which asks the user to enter login details and included an image of train. Verification is done whether the entered username and password match with the records in our data-base. If they match, we proceed further. Else, Informs the user to re-enter username and password.

4.5.1 Detailed Description

The LoginDialog class.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 LoginDialog()

```
LoginDialog::LoginDialog (
            QWidget * parent = nullptr )  [explicit]
```

LoginDialog - This constructor displays the home-page, which asks the user to enter login details and included an image of train. Verification is done whether the entered username and password match with the records in our data-base. If they match, we proceed further. Else, Informs the user to re-enter username and password.

Parameters

parent	- variable (of type QWidget ∗) which represents the Dialog.

The documentation for this class was generated from the following files:

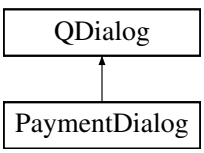
- BookMyJourney/Headers/loginDialog.h
- BookMyJourney/Sources/loginDialog.cpp

4.6 PaymentDialog Class Reference

The PaymentDialog class.

```
#include <paymentDialog.h>
```

Inheritance diagram for PaymentDialog:



Public Member Functions

- PaymentDialog (QWidget ∗parent=nullptr)

  PaymentDialog - This constructor displays the User interface which includes the payment details. We will have a timer of 20 seconds, before which the user has to decide whether he/she is willing to book tickets. After 10 seconds, the session expires and asks the user to re-book tickets. As user confirms the payment, we assume that the user has paid for it and we reserve the tickets for the user.

4.6.1 Detailed Description

The PaymentDialog class.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 PaymentDialog()

```
PaymentDialog::PaymentDialog (
            QWidget * parent = nullptr )  [explicit]
```

PaymentDialog - This constructor displays the User interface which includes the payment details. We will have a timer of 20 seconds, before which the user has to decide whether he/she is willing to book tickets. After 10 seconds, the session expires and asks the user to re-book tickets. As user confirms the payment, we assume that the user has paid for it and we reserve the tickets for the user.

Parameters

parent	- variable (of type QWidget $*$) which represents the Dialog.

The documentation for this class was generated from the following files:

- BookMyJourney/Headers/paymentDialog.h
- BookMyJourney/Sources/paymentDialog.cpp

4.7 seatmatrix Struct Reference

The seatmatrix class.

Public Attributes

- char seats [50]

  seats[i] - Denotes the status of seat with ID i (whether the seat was booked already or not).

4.7.1 Detailed Description

The seatmatrix class.

The documentation for this struct was generated from the following file:

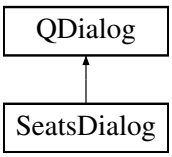
- server.cpp

4.8 SeatsDialog Class Reference

The SeatsDialog class.

`#include <seatsDialog.h>`

Inheritance diagram for SeatsDialog:



Public Member Functions

- SeatsDialog (QWidget *parent=nullptr)

  SeatsDialog - This constructor displays the seat matrix of appropriate train (of particular date) to the user. Seats which are already booked are grayed-out. All the concurrency revolves around this dialog. We have used Binary Semaphores to mutually exclude seats selection.

4.8.1 Detailed Description

The SeatsDialog class.

4.8.2 Constructor & Destructor Documentation

4.8.2.1 SeatsDialog()

```
SeatsDialog::SeatsDialog (
            QWidget * parent = nullptr )  [explicit]
```

SeatsDialog - This constructor displays the seat matrix of appropriate train (of particular date) to the user. Seats which are already booked are grayed-out. All the concurrency revolves around this dialog. We have used Binary Semaphores to mutually exclude seats selection.

Parameters

parent	- variable (of type QWidget *) which represents the Dialog.

The documentation for this class was generated from the following files:

- BookMyJourney/Headers/seatsDialog.h
- BookMyJourney/Sources/seatsDialog.cpp