Railway Berth Booking System

- User Registration And Authentication
- User-Friendly Text Interface
- User Login from Terminals of different machines
- Multithreaded Platform to Enhance Scalability
- Well Documented Program for Easy Modification

Given: [1] ReserveSys (Railway/Hotel/Bus) platform

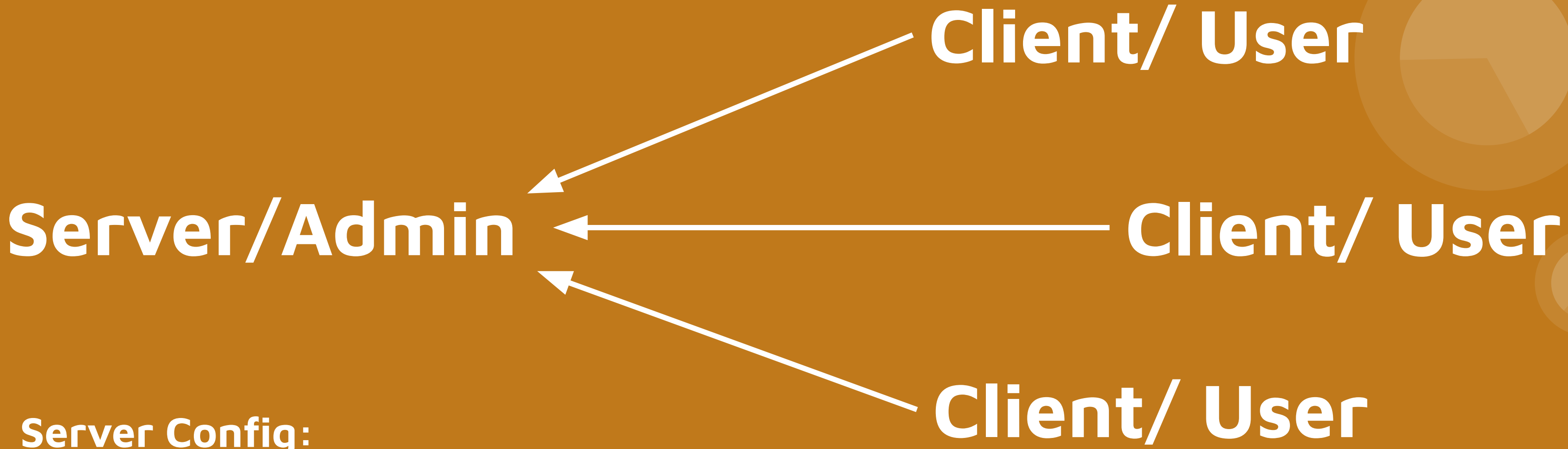
Develop a reservation platform, (hotel rooms/bus seats/railway berths booking system), where rooms/seats are offered based on user requests

Probable Features of the system:

- Define reservation policy
- Display various deals and packages
- Create a dynamic pricing reservation system
- Seats/Rooms allocation policies (e.g. aged persons will be allocated front rows/lower floors)
- Text-based user interface & availability matrix visualization, report & ticket generation

The system satisfy the following requirements:

- Enables user login from different terminals from different physical machines
- Enables User registration and authentication (using some hash based password)
- A user-friendly text interface
- A multithreaded platform to enable scalability
- Provide a well documented header file with set of well defined APIs/function interface so that any other reservation scenario may be implemented with minor modification



Server Config:

✷ User Management:
Create, Authenticate, Delete

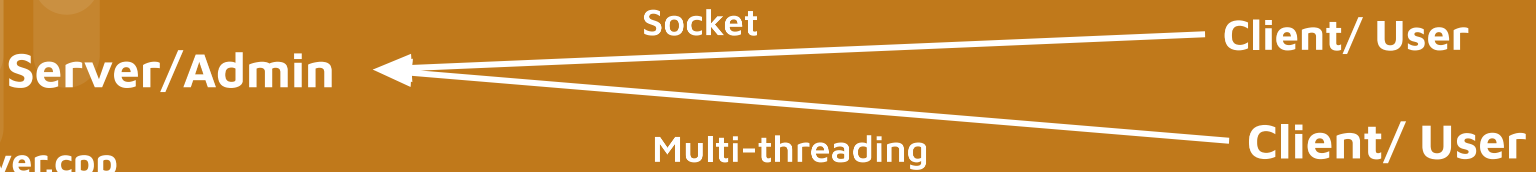
✷ Train Management:
Add, Remove, Display

✷ Reservation Management:
Book-Ticket, Cancel-Ticket, Show-Coach

Client Config:

✷ Account Management:
Register, Login, Delete-Account

✷ Ticket Management:
Reserve, Cancel, Status, Enquire-Trains



File: server.cpp
Server: (Objects)
✷ User: (Functions)
Create(Mobile, Password), Authenticate(Mobile, Password), Delete(Mobile, Password)
✷ Train: (Functions)
Add(Train-no, Source, Destination), Remove(Train-no, Source, Destination), Display() -Return: trn-no, src,dest, st-time, end-time, price[dynamic per booked seats]
✷ Reservation: (Functions)[++Policy-Later]
(ticket.txt)Book-Ticket(user,train), (cancel.txt)Cancel-Ticket(user,train), [update empty seats, refund]
Show-Coach(train)

File: client.cpp
Client: (Objects)`
✷ Account: (Functions)
Register(Mobile, Password), Login(Mobile, Password), Delete-Account(Mobile, Password)
✷ Ticket: (Functions)
Reserve(Mobile, Password,trn-no, src, dest), Cancel(Mobile, Password,ticket-id), Status(Mobile, Password,ticket-id), Enquire-Trains(trn-no)

Socket:
Create socket, bind and listen server, user sending request, accept requests(divide across threads), close client, local ip address and routers