Chat Portal

C++ PROJECT

Regd no. 184415

What is it based on ???

- A pure client-server based chat application
- So, of course, a **Client** side & a **Server** side program
- By chat, I mean **chat room**
- A single **group** chat for whole of MDH
- Server PC is assumed to be always-on
- In this presentation, I am taking 192.168.34.XX to be the Server.

Disclaimer

- It does have some issues
- Would have taken some more time to rectify them
- Many features not included
- Only to promote ease

Mostly Used OOP Concepts

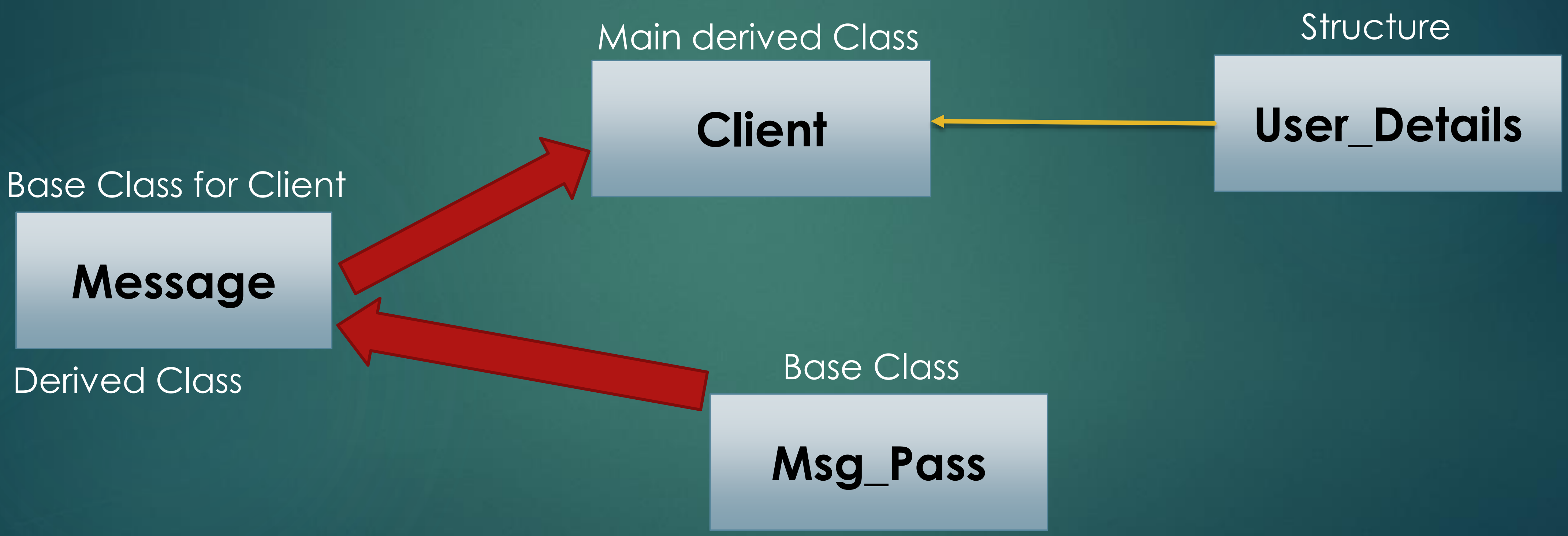
- Multiple Classes (with ctor & destructor)
- Inheritance
- Static data members and functions
- Polymorphism
- Aggregation & Composition
- Template
- Exception Handling
- Virtual Function
- Abstract Classes
- Namespaces
- File Handling

Extra concepts used throughout

- POSIX Threads
- Socket Programming
- Signal Handling

Chat Portal V1.0

Class Diagram (Client Side)



Main derived Class

Client

Structure

User_Details

Base Class for Client

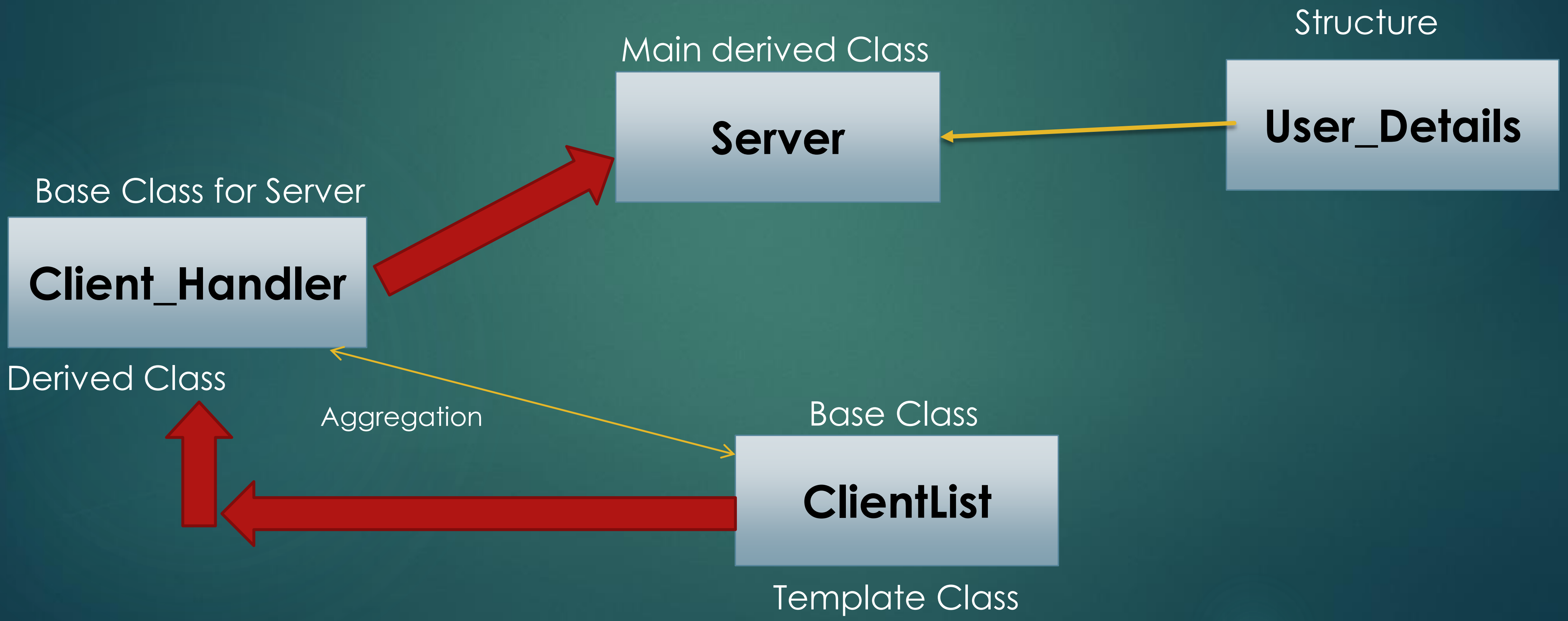
Message

Derived Class

Base Class

Msg_Pass

Class Diagram (Server Side)

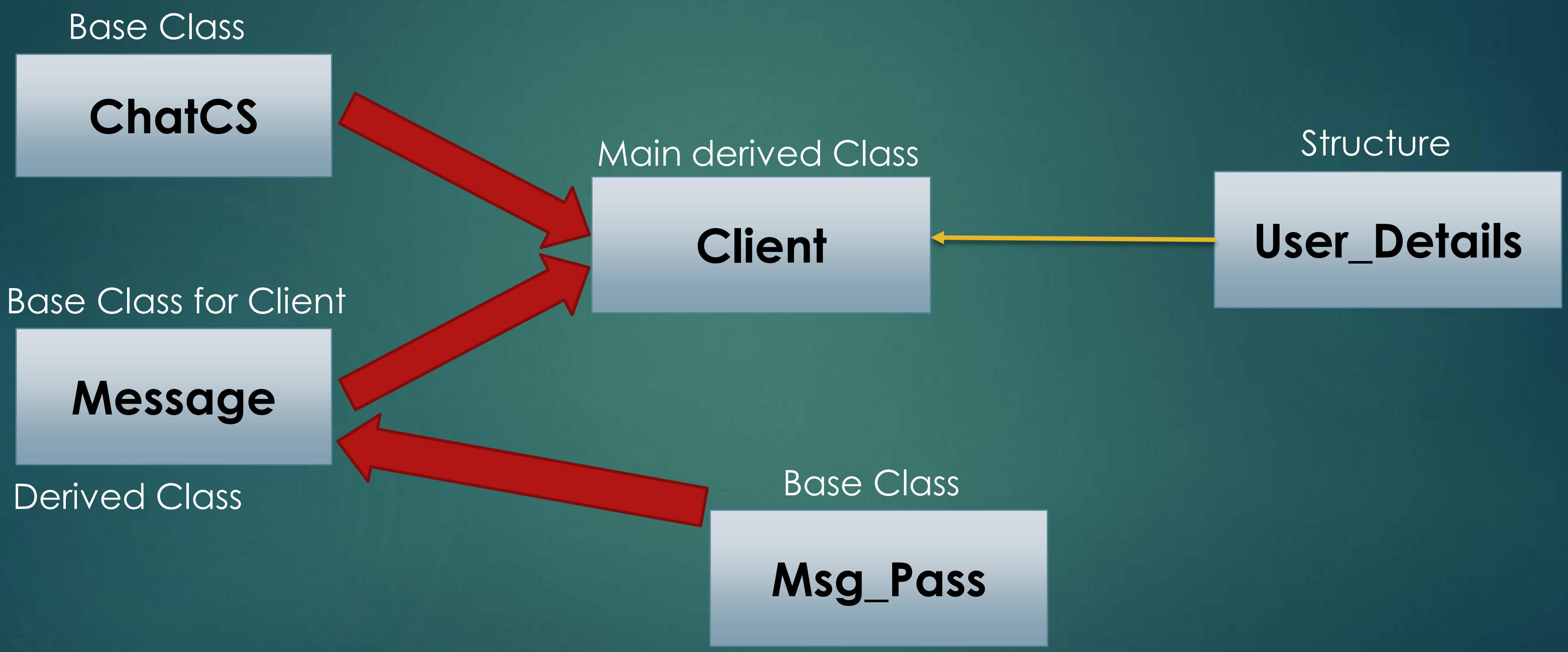


Concepts Covered

- Multiple Classes
- New & Delete operators
- Inheritance & its types
- Polymorphism (Operator Overloading)
- Static varibales and functions
- Nesting of functions
- Ctors & Destructors
- Ctor Overloading
- Abstract Classes
- Aggregation

Chat Portal V2.0

Class Diagram (Client Side)



Base Class

ChatCS

Main derived Class

Client

Structure

User_Details

Base Class for Client

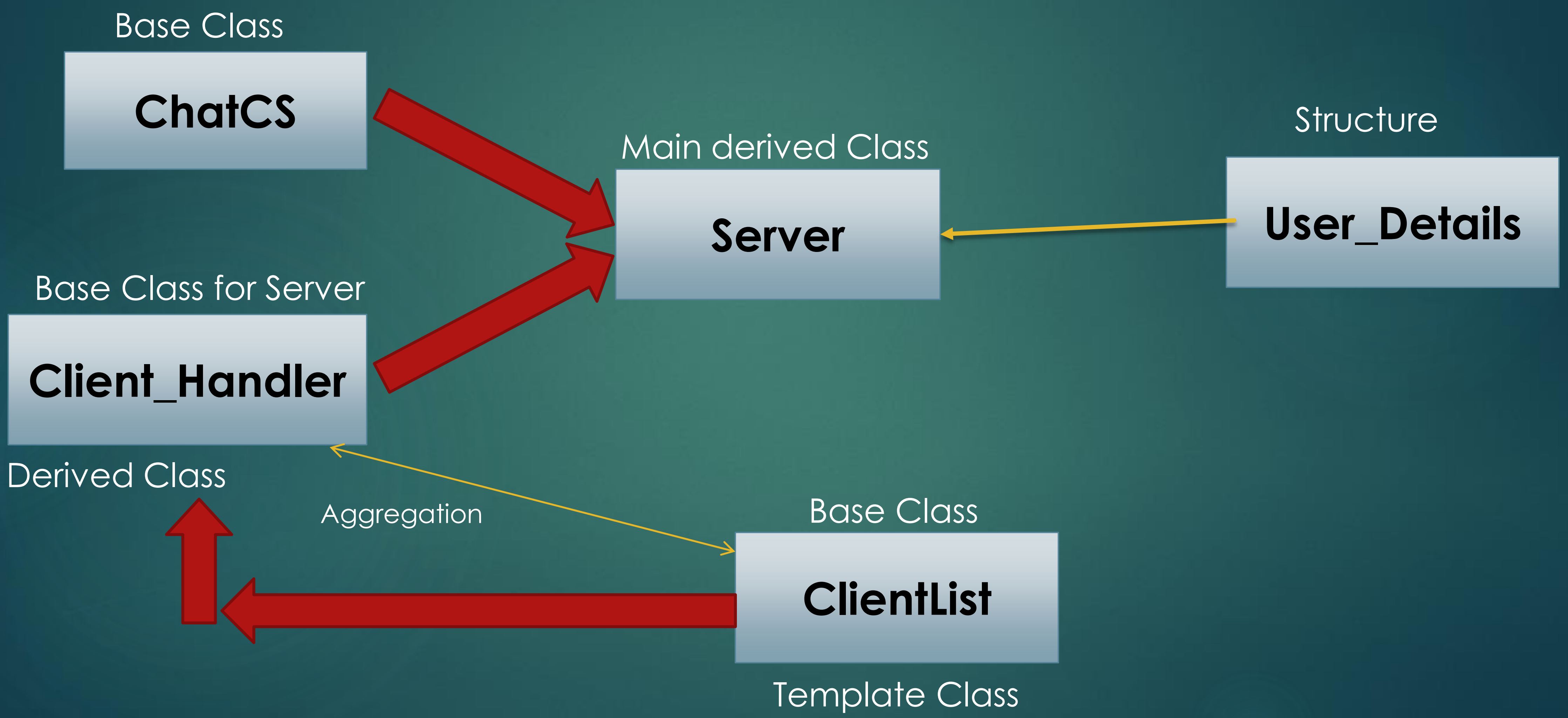
Message

Derived Class

Base Class

Msg_Pass

Class Diagram (Server Side)



Base Class

ChatCS

Main derived Class

Server

Structure

User_Details

Base Class for Server

Client_Handler

Derived Class

Aggregation

Base Class

ClientList

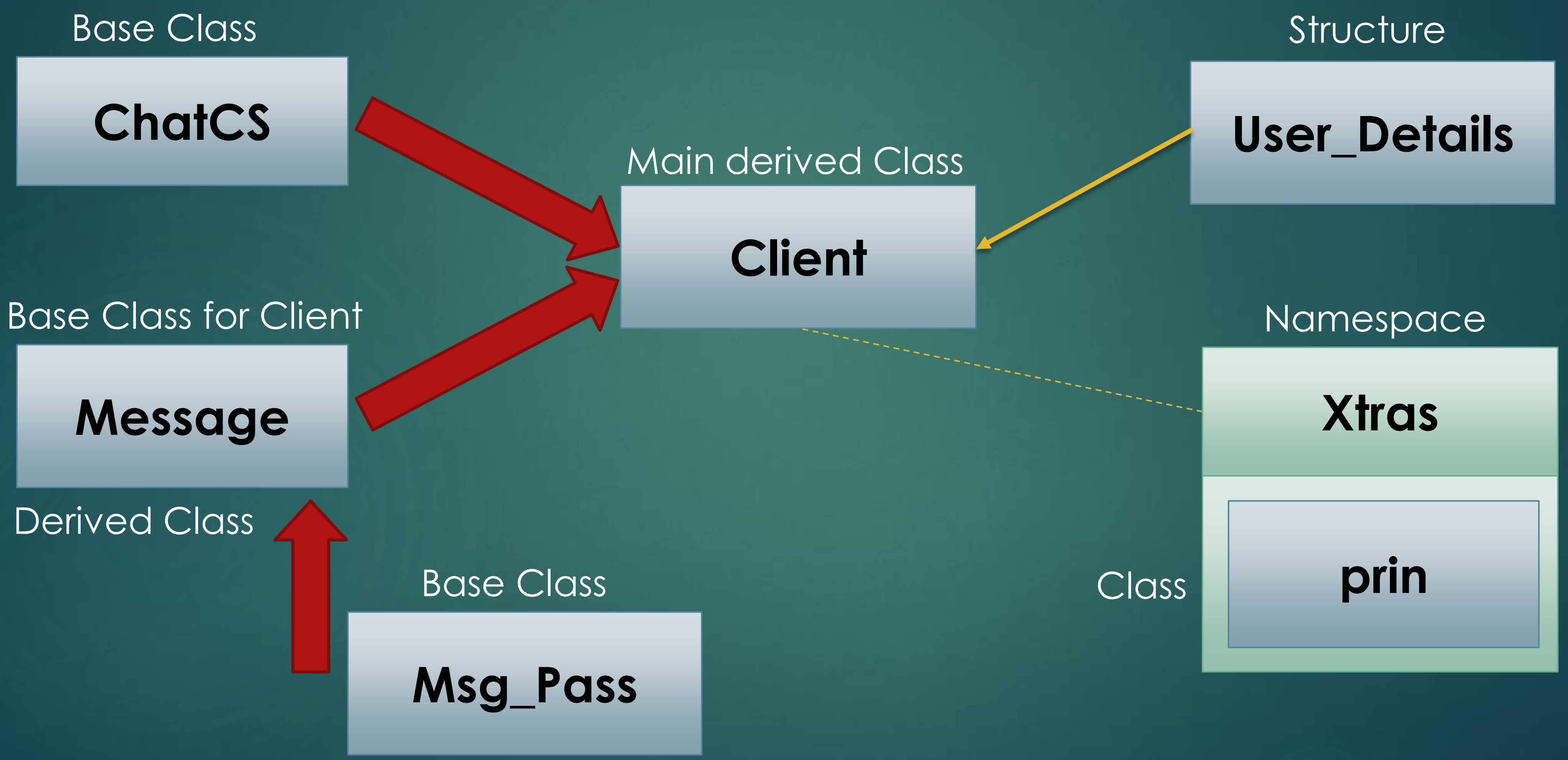
Template Class

More Concepts Covered

- Virtual Functions
- Abstract Classes
- Templates
- Generic Programming
- Exception Handling

Chat Portal V3.0

Class Diagram (Client Side)



Base Class

ChatCS

Main derived Class

Client

Structure

User_Details

Base Class for Client

Message

Derived Class

Namespace

Xtras

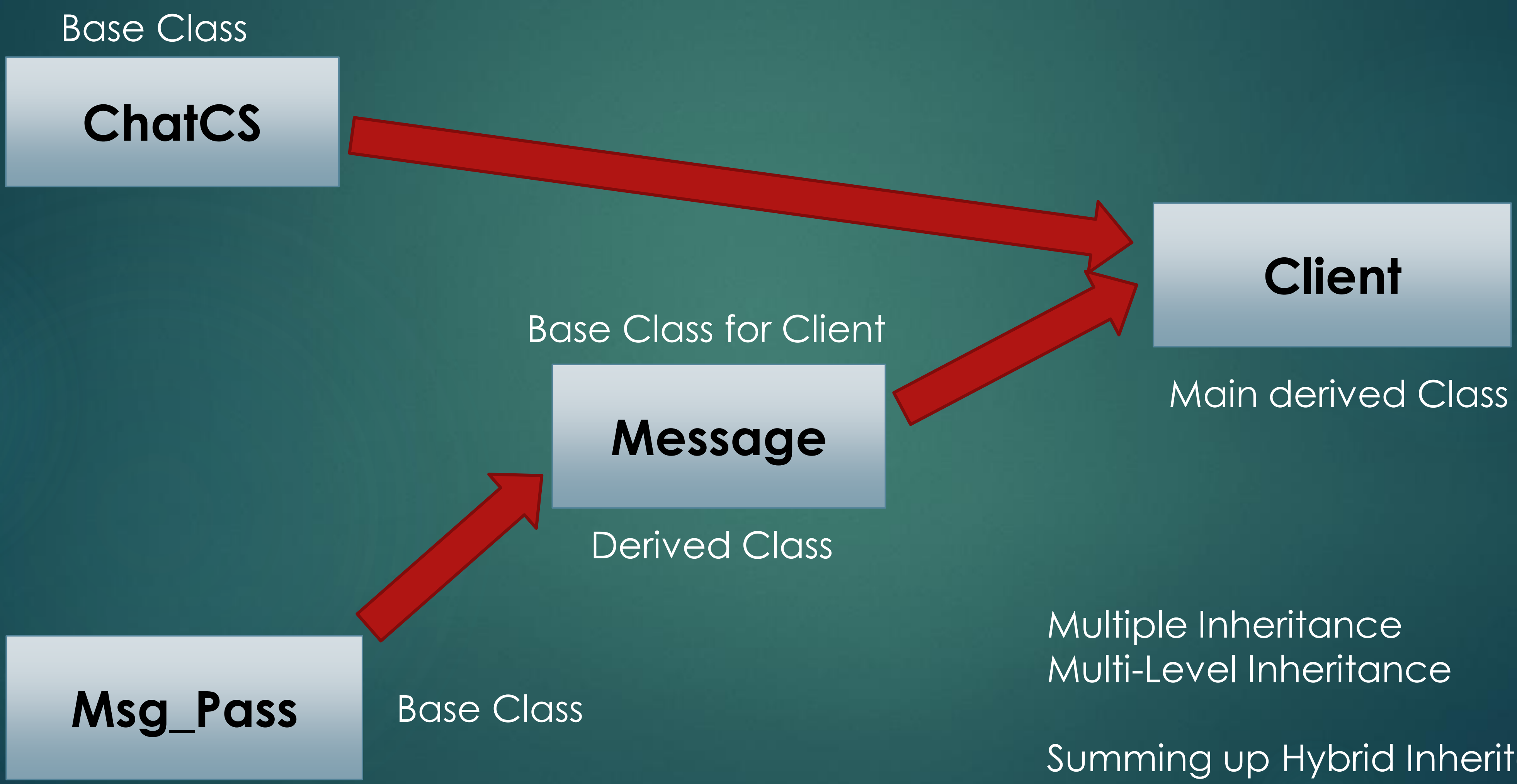
Class

prin

Base Class

Msg_Pass

Inheritance(Client Side)



Multiple Inheritance
Multi-Level Inheritance

Summing up Hybrid Inheritance

More Concepts Covered

- Nameless object
- File Handling
- Namespaces

Msg_pass

- Data members : -
- Static int sock_id – to store the active client socket
- Static int **flag** – to mark the end the send and recv function

- Member methods : -
- Msg_pass() – Default **ctor**
- Msg_pass(int s) – Parameterized **ctor** (socket passed as parameter)
- Static void Ctrl_C_handler(int **sig**)
- Static **Recv_msg_handler**()
- Static **Send_msg_handler**()
- ~Msg_pass()

Message : public Msg_pass

- Data members : -
- None

- Member methods : -
- Message(int sock) – Parameterized **ctor** (socket passed as parameter)
- Void Chat() – Deploys **threads** for sending and receiving messeges
- ~Message()

ChatCS (Abstract Class)

- Data members : -
- None

- Member methods : -
- Virtual void Set_IP()
- Virtual void Estd_connect()
- Virtual void Init_chat()

Client : public Message, ChatCS

- Data members : -
- Int error – serves as error flag
- User_Details info – as name suggests
- **Struct sockaddr_in** server_info , client_info – provided by the system
- Int size_server , size_client

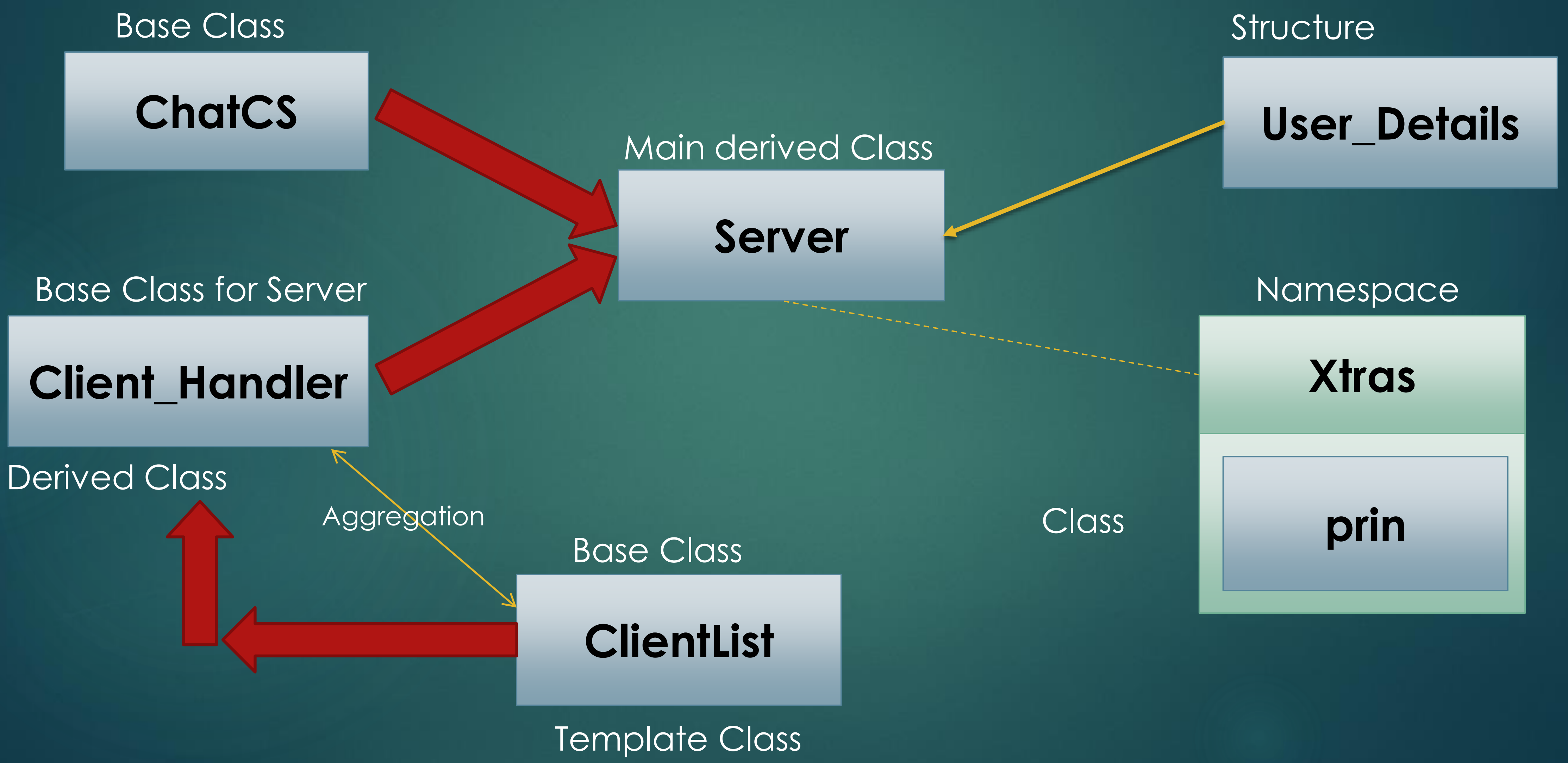
- Member methods : -
- Client(User_Details sm, int sock = -1)
- Void Set_IP()
- Void Estd_connect()
- Void Init_chat()
- ~Client

Struct User_Details

- Data members : -
- Name
- Username
- Password
- DOB

- Member methods : -
- Register()
- SignIn()
- Display_Profile()
- Edit_Profile()
- **Operator Overloading** of “=“

Class Diagram (Server Side)



Base Class

ChatCS

Main derived Class

Server

Structure

User_Details

Base Class for Server

Client_Handler

Derived Class

Namespace

Xtras

prin

Class

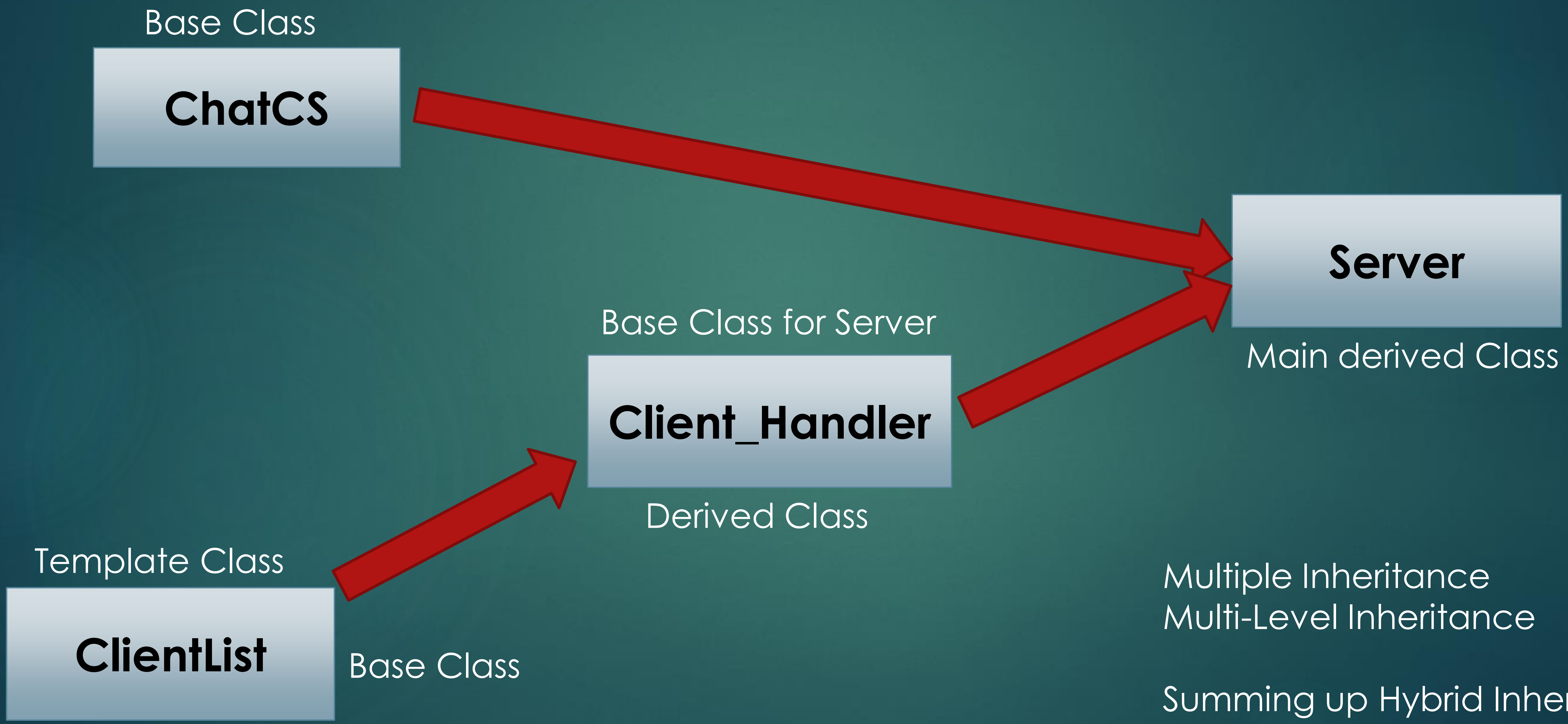
Aggregation

Base Class

ClientList

Template Class

Inheritance(Server Side)



Base Class

ChatCS

Server

Main derived Class

Base Class for Server

Client_Handler

Derived Class

Template Class

ClientList

Base Class

Multiple Inheritance
Multi-Level Inheritance

Summing up Hybrid Inheritance

Template ClientList <class T>

- Data members : -
- Static int active
- Int socket
- ClientList<T> *prev, *link
- T info (we are taking T as User_Details)

- Member methods : -
- Operator Overloading of “new”
- Operator Overloading of “delete”
- Operator Overloading of “=“
- newNode(int sock, char IP[16]) – adding a new node to the linked list

Client_Handler : public ClientList

- Data members : -
- 2 ClientList<User_Details> pointers

- Member methods : -
- Default ctor()
- Remove_Clients()
- Send_to_all_clients()
- Handler()
- Destructor

Server : public Client_Handler, ChatCS

- Data members : -
- Int error – serves as error flag
- User_Details info
- Struct sockaddr_in server_info , client_info
- Int size_server , size_client

- Member methods : -
- Server(int sock = -1)
- Void Set_IP()
- Void Estd_connect()
- Void Init_chat()
- ~Server()

Namespace Xtras

- Classes : -
- Prin – Prints welcome message

- Methods : -
- CreateSock() – creates socket

Exception Handling

- Used to detect and deal with system errors
- Sockets and Threads error
- Wrong input detection

File Handling

- Used to store information of the user
- Writing to file in case of Sign Up or Edit Profile
- Retriving from file in case of Sign In
- File name “<RegdNo>”

Thank You