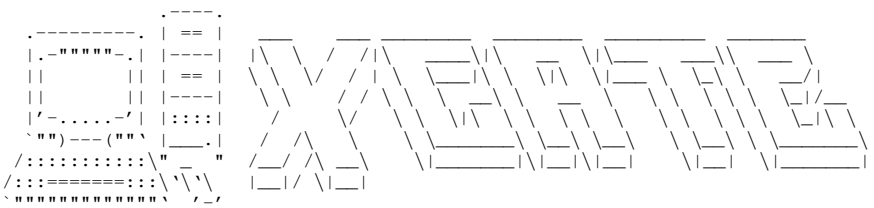
Tugas Akhir Warnet xGate

1.0

Generated by Doxygen 1.10.0

Chapter 1

Project Tugas Akhir Semester 2



Project tugas akhir TI E '23 UNESA\ Membuat mockup sistem manajemen internet cafe `Link GitHub Repo`

1.1 Installation

Use the GNU g++ compiler to compile main.cpp

```
g++ main -o main; ./main
```

1.2 Anggota Kelompok

- [x] **23051204157** Mohammad Mujib Nur Rohman
- [x] **23051204162** Gabriel Michael Tanu Wijaya
- [x] **23051204180** Asyary Raihan Haryono

1.3 Credit

`SHA-256 C++ Implementation by okdshin`

1.3.1 Sidenote

Default login admin dengan pin 123456

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Class Documentation

4.1 ComputerTree Struct Reference

ComputerTree struct Struct for computer tree, contains jumlahChild, nama, jenis, isUsed, next node, and child node.

Public Attributes

- int **jumlahChild** = 0
- string **nama**
- string **jenis**
- bool **isUsed** {0}
- ComputerTree * **next** {NULL}
- ComputerTree * **child** {NULL}

4.1.1 Detailed Description

ComputerTree struct Struct for computer tree, contains jumlahChild, nama, jenis, isUsed, next node, and child node.

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

- main.cpp

4.2 NodeQueue Struct Reference

NodeQueue struct Struct for queue node, contains nama and next node.

Public Attributes

- string **nama**
- NodeQueue * **next** {NULL}

4.2.1 Detailed Description

NodeQueue struct Struct for queue node, contains nama and next node.

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

- main.cpp

4.3 NodeUser Struct Reference

NodeUser struct Struct for user node, contains User data and next node.

Public Attributes

- User **data**
- NodeUser * **next** {NULL}

4.3.1 Detailed Description

NodeUser struct Struct for user node, contains User data and next node.

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

- main.cpp

4.4 User Struct Reference

User struct Struct for user data, contains nama, username, password, level, and hasBilling.

Public Attributes

- string **nama**
- string **username**
- string **password**
- string **level** = "user"
- bool **hasBilling** {0}

4.4.1 Detailed Description

User struct Struct for user data, contains nama, username, password, level, and hasBilling.

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

- main.cpp

Chapter 5

File Documentation

5.1 main.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <thread>
#include <regex>
#include <cmath>
#include <csignal>
#include <iomanip>
#include "include/utils.h"
#include "include/sha256.h"
```

Classes

- struct User

  User struct Struct for user data, contains nama, username, password, level, and hasBilling.
- struct NodeUser

  NodeUser struct Struct for user node, contains User data and next node.
- struct NodeQueue

  NodeQueue struct Struct for queue node, contains nama and next node.
- struct ComputerTree

  ComputerTree struct Struct for computer tree, contains jumlahChild, nama, jenis, isUsed, next node, and child node.

Enumerations

- enum Menu { **MAIN_MENU** , **ADMIN_MENU** , **USER_MENU** }

  Enum for menus.
- enum InputType { NAME , USERNAME , PASSWORD }

  Enum for input types.

Functions

- void **quit** ()

  Function to quit Quits the program gracefully.
- void menu (Menu dest)

  Function to show menus Shows the menus based on the destination using menu enum.
- void **updateUserDB** ()

  Function to update userDB Updates the user database.
- void **updatePCDB** ()

  Function to update PCDB Updates the PC database.
- char inputHandler ()

  Fancy smanchy function to handle inputs Handles CTRL+C, CTRL+D, CTRL+X, CTRL+Z.
- string inputHandlerStr (InputType type)

  Even more fancy schmancies, woohoo Handles input for strings with input type and regex.
- void **showPCData** ()

  Function to show PC data Shows all the available PCs.
- void **deleteQueue** ()

  Function to delete queue Deletes the first queue.
- void **konfirmasiBilling** ()

  Function to confirm billing Confirms the billing by admin.
- void treatAngka (double saldo, string ∗saldoStr, int ∗desimal)

  Function to treat angka Treats the number to be formatted with dots and 2 decimal places.
- void **errorHandler** (string err)

  Error handler Handles error messages.
- int **hashFunction** (string key)

  Hash function Hash function for hashing the key using custom multiplicative fibonacci hashing function.
- void addQueue (ComputerTree ∗pc)

  Add queue Adds the queue to the queue.txt file.
- void **pesanPC** ()

  Pesan PC Function to order a PC.
- void **addPC** ()

  Function to add PC Adds a PC to the server.
- string hashPass (string str)

  Function to hash string Hashes the string using sha256.
- User **userValidation** (string username)

  Function to validate user Validates the user using the username and hashmap with closed addressing collision handling.
- void daftar (string nama="", string username="")

  Function to daftar user Registers a user to the db and hashmap.
- void login (string username="")

  Function to login It says it in the name.
- void **readDB** ()

  Function to read database Reads the database from the user.txt and pc.txt files, and queues from queue.txt.
- void **loadingScr** ()

  Function to show loading screen Shows the loading screen with a spinner.
- void **greet** ()

  Function to greet Greets the user with ascii art banner.
- void **init** ()

  Initialize the program Initializes the program by reading the database, initializing the hashmap, and preparing the chef to cook (menyala abangkuh)
- int main ()

  Main function If you're asking what this is, I don't know what to tell you bud.

Variables

- struct NodeQueue * **headQueue**
- struct NodeQueue * **tailQueue**
- bool **isQuit** = false
- bool **doneLoading** = false
- bool **doneReading** = false
- int **totalUser** = 0
- int **totalRouter** = 0
- int **totalQueue** = 0
- const int **hashMapSize** = 2048
- NodeUser * **hashMapUser**
- User **currentUser**
- ComputerTree * **server** = new ComputerTree

5.1.1 Enumeration Type Documentation

5.1.1.1 InputType

```
enum InputType
```

Enum for input types.

Enumerator

NAME	name can contain anything, A-Za-z0-9 and \s whitespace
USERNAME	username can only contain A-Za-z0-9
PASSWORD	password can contain anything, ^[\x20-\x7E]+$

5.1.2 Function Documentation

5.1.2.1 addQueue()

```
void addQueue (
            ComputerTree * pc )
```

Add queue Adds the queue to the queue.txt file.

Parameters

pc	

5.1.2.2 daftar()

```
void daftar (
            string nama = "",
            string username = "" )
```

Function to daftar user Registers a user to the db and hashmap.

Parameters

nama	
username	

5.1.2.3 hashPass()

```
string hashPass (
            string str )
```

Function to hash string Hashes the string using sha256.

Parameters

str	

Returns

5.1.2.4 inputHandler()

```
char inputHandler ( )
```

Fancy smanchy function to handle inputs Handles CTRL+C, CTRL+D, CTRL+X, CTRL+Z.

Returns

5.1.2.5 inputHandlerStr()

```
string inputHandlerStr (
            InputType type )
```

Even more fancy schmancies, woohoo Handles input for strings with input type and regex.

Parameters

type	

Returns

5.1.2.6 login()

```
void login (
            string username = "" )
```

Function to login It says it in the name.

Parameters

username	

5.1.2.7 main()

```
int main ( )
```

Main function If you're asking what this is, I don't know what to tell you bud.

Returns

5.1.2.8 menu()

```
void menu (
            Menu dest )
```

Function to show menus Shows the menus based on the destination using menu enum.

Parameters

dest	

5.1.2.9 treatAngka()

```
void treatAngka (
            double saldo,
            string * saldoStr,
            int * desimal )
```

Function to treat angka Treats the number to be formatted with dots and 2 decimal places.

Parameters

saldo	
saldoStr	
desimal	

Index