**D’MOV1E SEBAGAI APLIKASI MANAJEMEN TIKET BIOSKOP BERBASIS DESKTOP**

Diajukan Sebagai Tugas Mata Kuliah

Struktur Data



Anggota Kelompok:

1. Putu Widyantara Artanta Wibawa (2108561005)
2. Putu Putri Pratiwi (2108561010)
3. Gede Krisnawa Sandhya Wandhana (2108561017)

**PRODI INFORMATIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS UDAYANA**

**BUKIT JIMBARAN**

**2021**

**DAFTAR ISI**

Cover i

Daftar Isi ii

BAB I. PENDAHULUAN 1

1. Latar Belakang 2
2. Rumusan Masalah 2
3. Tujuan 2
4. Manfaat 2

BAB II. ANALISIS DAN RANCANGAN 3

1. Analisis 3
2. Rancangan 4

BAB III. IMPLEMENTASI DAN HASIL CAPTURE 5

1. Implementasi 5
2. Hasil Capture 52

BAB IV. PENUTUP 56

1. Kesimpulan 56

**BAB I**

**PENDAHULUAN**

1. **Latar Belakang**

Bioskop merupakan tempat yang mempertunjukkan film untuk dinikmati oleh masyarakat publik. Walaupun perkembangan teknologi kian pesat dimana masyarakat saat ini dapat menikmati sebuah film melalui siaran televisi maupun laptop atau *handphone* secara online, keberadaan bioskop masih banyak diminati. Bahkan penjualan tiket bioskop kerap meningkat pesat ketika dirilisnya suatu film baru atau ketika musim liburan tiba.

Untuk menikmati film dalam bioskop, pelanggan diharuskan membeli tiket di loket bioskop yang akan dituju. Dalam pembelian tiket, pelanggan diharuskan mengantri. Selanjutnya pelanggan dapat memilih film, jadwal tayang, dan tempat duduk yang tersedia, pelanggan melakukan pembayaran langsung pada loket bioskop. Namun proses pembelian pada bioskop ini sering menimbulkan permasalahan seperti antrian panjang pada loket sehingga mengakibatkan pembatalan tiket karena jam tayang yang terlewat. Selain itu karena antrian yang panjang, tidak sedikit pula pelanggan yang kehabisan tiket sehingga tidak bisa menikmati film. Pandemi Covid-19 juga turut andil dalam permasalahan pada bioskop. Pihak bioskop harus membatasi jumlah pengunjung untuk meminimalisir penularan virus Covid-19. Hal tersebut menyebabkan pelanggan menjadi kesulitan dalam membeli tiket di bioskop.

Dari permasalahan tersebut, diperlukan solusi berupa pembelian tiket bioskop via *online* melalui suatu aplikasi agar pelanggan dapat membeli tiket bioskop dengan nyaman tanpa adanya antrian panjang pada loket bioskop, ketinggalan jam tayang, dan meminimalisir kerumunan pada bioskop di tengah pandemic Covid-19. Oleh karena itu, diperlukan sebuah aplikasi yang mampu untuk melakukan manajemen pembelian tiket di bioskop, sehingga proses pembelian tiket, pemilihan kursi, dan pembayaran dapat berlangsung dengan mudah.

1. **Rumusan Masalah**

Berdasarkan latar belakang diatas, dapat ditarik rumusan masalah sebagai berikut:

1. Bagaimana aplikasi manajemen pembelian tiket bioskop tersebut dibuat?
2. Bagaimana tampilan dari aplikasi manajemen pembelian tiket bioskop tersebut?
3. **Tujuan**

Adapun tujuan dibuatnya aplikasi ini selain memenuhi tugas akhir mata kuliah Struktur Data adalah untuk menciptakan suatu solusi dari permasalahan pembelian tiket bioskop secara langsung.

1. **Manfaat**

Pembuatan aplikasi ini akan memberikan manfaat kepada pelanggan maupun pihak bioskop karena proses pemesanan tiket ini bersifat praktis dan tidak menimbulkan antrian panjang pada bioskop**.**

**BAB II**

**ANALISIS DAN RANCANGAN**

* 1. **Algoritma**

Berdasarkan permasalahan yang ada, dapat dianalisis dan dibuatkan sebuah alur algoritma pemecahan masalah.

1. Masukkan username dan password
2. Jika username dan password sesuai maka masuk ke dashboard

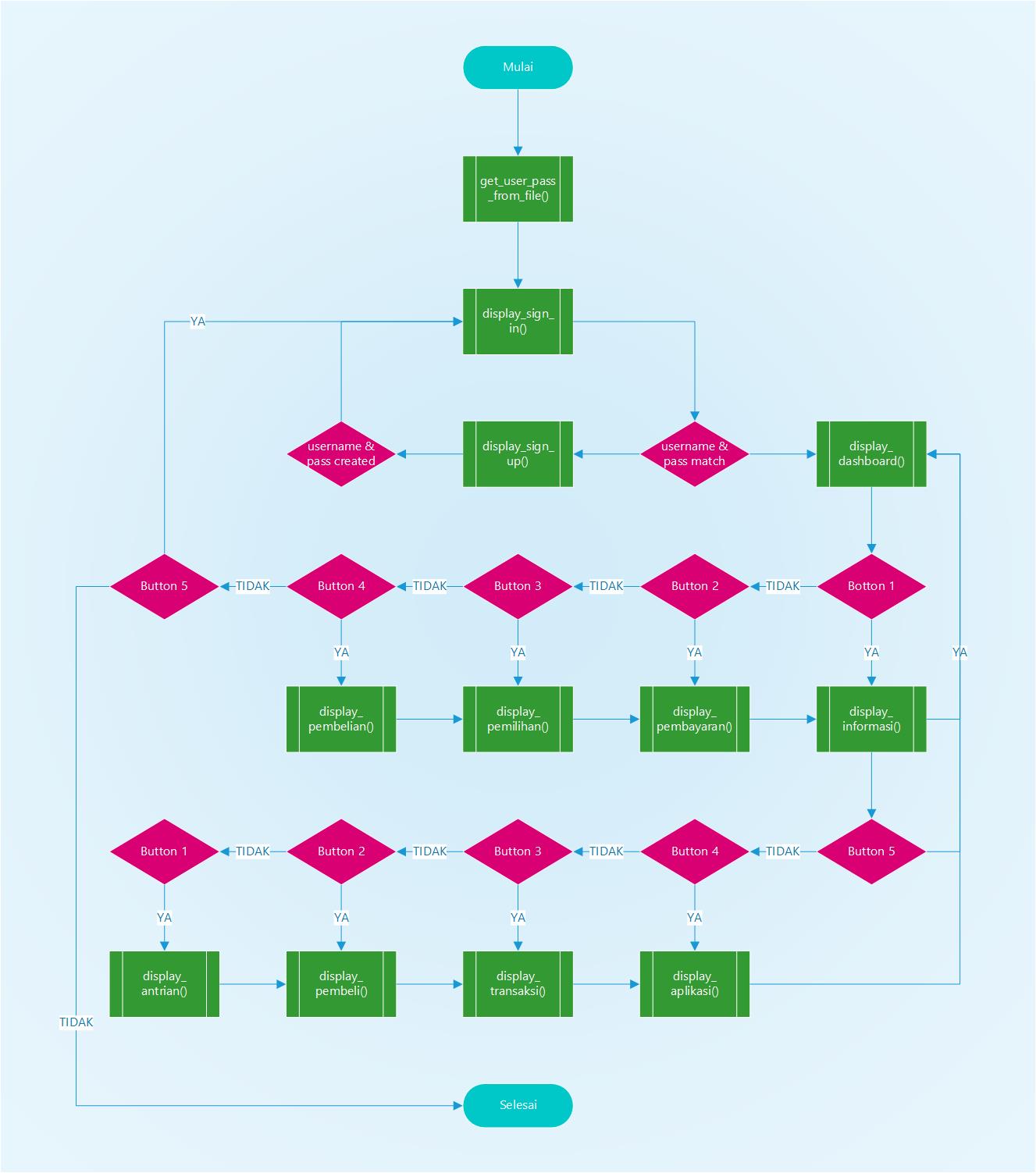
2.1 Jika tidak buat username dan password baru

1. Masuk ke menu pembelian
2. Masukkan nama pembeli, jumlah tiket, nama film, dan waktu film
3. Jika seluruh informasi telah sesuai masuk ke menu pemilihan
   1. Jika tidak tampilkan pesan error
4. Pilih kursi sesuai dengan jumlah tiket yang dipesan
5. Jika seluruh pilihan kursi telah sesuai masuk ke menu pembayaran

7.1 Jika tidak tampilkan pesan error

1. Masukkan jumlah uang yang dibayar
2. Jika jumlah uang yang dibayar telah sesuai maka konfirmasi pembayaran
   1. Jika tidak tampilkan pesan error
3. Masuk ke menu informasi
4. Masuk ke menu informasi antrian untuk melihat antrian pemilihan kursi dan antrian pembayaran
5. Masuk ke menu informasi pembeli untuk melihat pembeli terbaik
6. Masuk ke menu informasi transaksi untuk melihat transaksi terakhir
7. Masuk ke menu informasi aplikasi untuk melihat informasi tentang aplikasi
   1. **Rancangan**

Berdasarkan analisis pemecahan masalah dengan alur algoritma yang sudah ada, dapat dibuat sebuah *flowchart* atau diagram alir yang menggambarkan rancangan dari program.



**BAB III**

**IMPLEMENTASI DAN HASIL CAPTURE**

1. **Implementasi**

Berikut adalah implementasi dari flowchart atau diagram alir program manajemen tiket bioskop ke dalam source code bahasa pemrograman C.

**main.c**

|  |
| --- |
| #include "lib/d\_mov1e.h"  int main(){  get\_user\_pass\_from\_file();  display\_sign\_in();  return 0;  } |

**d\_mov1e.c**

|  |
| --- |
| #include "d\_mov1e.h"  // LINKED LIST  UserInfo \*User, \*Head;  // QUEUE  Queue \*qUser, \*qFront, \*qFrontPay;;  // STACK  Stack \*sChair, \*sTop, \*sTopChair;  // BST  BST \*bRoot, \*bNode;  char FilmChair[16][3] ={  "A1","A2","A3","A4",  "B1","B2","B3","B4",  "C1","C2", "C3", "C4",  "D1", "D2", "D3", "D4"  };  char FilmName[4][50] = {  "KKN Desa Penari",  "The Throne",  "Ready Player One",  "Stand by Me Doraemon 2"  };  char FilmTime[4][3][20] = {  {  "08.00-10.00",  "10.00-12.00",  "13.30-15.30"  },  {  "08.30-10.00",  "11.30-13.00",  "13.00-14.30"  },  {  "10.30-11.30",  "11.30-12.30",  "12.30-13.30"  },  {  "09.00-10.30",  "10.00-11.30",  "12.00-13.30"  }  };  char StudioTime[3][4][20] = {  {  "08.00-10.00",  "10.00-12.00",  "12.00-13.30",  "13.30-15.30"  },  {  "08.30-10.00",  "10.00-11.30",  "11.30-13.00",  "13.00-14.30"  },  {  "09.00-10.30",  "10.30-11.30",  "11.30-12.30",  "12.30-13.30"  }  };  char StudioChair[3][4][16];  int TicketPrice[4] = {80000, 60000, 65000, 70000};  int AvailableChair[3][4];  int MoneyCurrency[7] = {100000, 50000, 20000, 10000, 5000, 2000, 1000};  bool admitLogin;  void reset\_string(char \*str, int len){  int i;  for (i = 0; i < len; i++){  str[i] = '\0';  }  }  void display\_info\_dialog(const gchar \*info, GtkWidget \*window){  GtkWidget \*dialog;  dialog = gtk\_message\_dialog\_new(GTK\_WINDOW(window),  GTK\_DIALOG\_DESTROY\_WITH\_PARENT,  GTK\_MESSAGE\_INFO,  GTK\_BUTTONS\_OK,  info);  gtk\_widget\_set\_name(dialog, "infoDialog");  gtk\_window\_set\_title(GTK\_WINDOW(dialog), "Information");  gtk\_dialog\_run(GTK\_DIALOG(dialog));  gtk\_widget\_destroy(dialog);  }  void display\_error\_dialog(const gchar \*error, GtkWidget \*window){  GtkWidget \*dialog;  dialog = gtk\_message\_dialog\_new(GTK\_WINDOW(window),  GTK\_DIALOG\_DESTROY\_WITH\_PARENT,  GTK\_MESSAGE\_ERROR,  GTK\_BUTTONS\_OK,  error);  gtk\_widget\_set\_name(dialog, "errorDialog");  gtk\_window\_set\_title(GTK\_WINDOW(dialog), "Error");  gtk\_dialog\_run(GTK\_DIALOG(dialog));  gtk\_widget\_destroy(dialog);  }  void display\_warn\_dialog(const gchar \*warn, GtkWidget \*window){  GtkWidget \*dialog;  dialog = gtk\_message\_dialog\_new(GTK\_WINDOW(window),  GTK\_DIALOG\_DESTROY\_WITH\_PARENT,  GTK\_MESSAGE\_ERROR,  GTK\_BUTTONS\_YES\_NO,  warn);  gtk\_widget\_set\_name(dialog, "warnDialog");  gtk\_window\_set\_title(GTK\_WINDOW(dialog), "Warning");  gtk\_dialog\_run(GTK\_DIALOG(dialog));  gtk\_widget\_destroy(dialog);  }  void get\_user\_pass\_from\_file(){  char compUsr[100], compPwd[100];  reset\_string(compUsr, 100);  reset\_string(compPwd, 100);  Head = NULL;  FILE \*f;  f = fopen("data/user\_pass\_data.txt", "r");  char ch;  int idx = 1, counter = 0;  while(!feof(f)){  ch = fgetc(f);  if (ch == '|'){  counter = 0;  idx = 2;  }  else if (ch == '\n'){  User = malloc(sizeof(UserInfo));  strcpy(User->Username, compUsr);  strcpy(User->Password, compPwd);  User->Next = NULL;  if (Head == NULL){  Head = User;  }  else{  UserInfo \*prev;  prev = Head;  while(prev->Next != NULL){  prev = prev->Next;  }  prev->Next = User;  }  // RESET  reset\_string(compUsr, 100);  reset\_string(compPwd, 100);  counter = 0;  idx = 1;  }  else{  if (idx == 1){  compUsr[counter] = ch;  counter++;  }  else if(idx == 2){  compPwd[counter] = ch;  counter++;  }  }  }  fclose(f);  }  char\* do\_password\_hash(char \*password){  static char result[100];  reset\_string(result, 100);  SHA256\_CTX context;  unsigned char md[SHA256\_DIGEST\_LENGTH];  SHA256\_Init(&context);  SHA256\_Update(&context, (unsigned char \*)password, sizeof(password));  SHA256\_Final(md, &context);  int i = 0;  while(md[i] != '\0'){  int temp = md[i];  int modTemp = temp % 16;  if (modTemp < 10){  result[i] = modTemp + '0';  }  else{  modTemp -= 10;  result[i] = modTemp + 'a';  }  i++;  }  return result;  }  char \*timeToStr(struct tm \*timeInfo){  static const char weekDay[][10] = {  "Minggu", "Senin", "Selasa", "Rabu", "Kamis", "Jumat", "Sabtu"};  static const char monthDay[][15] = {  "Januari", "Februari", "Maret", "April", "Mei", "Juni",  "Juli", "Agustus", "September", "Oktober", "November", "Desember"};  static char timeResult[100];  reset\_string(timeResult, 100);  sprintf(timeResult, "%.2d:%.2d:%.2d - %s, %d %s %d",  timeInfo->tm\_hour,  timeInfo->tm\_min,  timeInfo->tm\_sec,  weekDay[timeInfo->tm\_wday],  timeInfo->tm\_mday,  monthDay[timeInfo->tm\_mon],  1900 + timeInfo->tm\_year);  return timeResult;  } |

**sign\_in.c**

|  |
| --- |
| #include "sign\_in.h"  // WINDOW  GtkWidget \*windowSignIn;  // INPUT  GtkWidget \*entryUsernameSignIn;  GtkWidget \*entryPasswordSignIn;  void check\_login(){  char pass[100], username[100], passHashing[100];  reset\_string(pass, 100);  reset\_string(username, 100);  reset\_string(passHashing, 100);  const gchar \*gtextUsername = gtk\_entry\_get\_text(GTK\_ENTRY(entryUsernameSignIn));  const gchar \*gtextPass = gtk\_entry\_get\_text(GTK\_ENTRY(entryPasswordSignIn));  sprintf(username, "%s", gtextUsername);  sprintf(pass, "%s", gtextPass);  strcpy(passHashing, pass);  strcpy(passHashing, do\_password\_hash(passHashing));  // CHECK  admitLogin = false;  UserInfo \*temp = Head;  while(temp != NULL){  if (strcmp(username, temp->Username) == 0 && strcmp(passHashing, temp->Password) == 0){  admitLogin = true;  break;  }  temp = temp->Next;  }  if (admitLogin){  display\_info\_dialog("Login Berhasil!", windowSignIn);  display\_dashboard(windowSignIn);  }  else{  display\_error\_dialog("Username atau Password Salah!", windowSignIn);  }  }  void display\_sign\_in(){  // WIDGET  GtkWidget \*layoutSignIn;  GtkWidget \*bgSignIn;  // ICON  GdkPixbuf \*iconSignIn;  // BUTTON  GtkWidget \*buttonSignIn;  GtkWidget \*buttonSignUp;  // LABEL  GtkWidget \*labelTitle;  GtkWidget \*labelContent;  GtkWidget \*labelUsername;  GtkWidget \*labelPassword;  GtkWidget \*labelNewAccount;  gtk\_init(NULL, NULL);  // WINDOW  windowSignIn = gtk\_window\_new(GTK\_WINDOW\_TOPLEVEL);  gtk\_window\_set\_title(GTK\_WINDOW(windowSignIn), "D'MOV1E");  gtk\_window\_set\_default\_size(GTK\_WINDOW(windowSignIn), 1280, 720);  gtk\_window\_set\_position(GTK\_WINDOW(windowSignIn), GTK\_WIN\_POS\_CENTER);  // LAYOUT  layoutSignIn = gtk\_layout\_new(NULL, NULL);  gtk\_container\_add(GTK\_CONTAINER (windowSignIn), layoutSignIn);  // ICON  iconSignIn = gdk\_pixbuf\_new\_from\_file("src/image/icon.png", NULL);  gtk\_window\_set\_icon(GTK\_WINDOW(windowSignIn), iconSignIn);  // BACKGROUND  bgSignIn = gtk\_image\_new\_from\_file("src/image/login.png");  gtk\_layout\_put(GTK\_LAYOUT(layoutSignIn), bgSignIn, 0, 0);  // LABEL  labelTitle = gtk\_label\_new("Welcome to D'MOV1E");  gtk\_layout\_put(GTK\_LAYOUT(layoutSignIn), labelTitle, 900, 100);  gtk\_widget\_set\_name(labelTitle, "labelTitle");  labelContent = gtk\_label\_new("Sign In");  gtk\_layout\_put(GTK\_LAYOUT(layoutSignIn), labelContent, 900, 150);  gtk\_widget\_set\_name(labelContent, "labelContent");  labelNewAccount = gtk\_label\_new("Belum punya akun?");  gtk\_layout\_put(GTK\_LAYOUT(layoutSignIn), labelNewAccount, 900, 500);  labelUsername = gtk\_label\_new("Username");  gtk\_layout\_put(GTK\_LAYOUT(layoutSignIn), labelUsername, 900, 270);  labelPassword = gtk\_label\_new("Password");  gtk\_layout\_put(GTK\_LAYOUT(layoutSignIn), labelPassword, 900, 355);  // ENTRY  entryUsernameSignIn = gtk\_entry\_new();  gtk\_widget\_set\_size\_request(entryUsernameSignIn, 250, 40);  gtk\_layout\_put(GTK\_LAYOUT(layoutSignIn), entryUsernameSignIn, 900, 300);  entryPasswordSignIn = gtk\_entry\_new();  gtk\_widget\_set\_size\_request(entryPasswordSignIn, 250, 40);  gtk\_layout\_put(GTK\_LAYOUT(layoutSignIn), entryPasswordSignIn, 900, 390);  gtk\_entry\_set\_visibility(GTK\_ENTRY(entryPasswordSignIn), false); // password hidden  gtk\_widget\_set\_name(entryPasswordSignIn, "entryPassSignIn");  // BUTTON  buttonSignIn = gtk\_button\_new\_with\_label("Login");  gtk\_layout\_put(GTK\_LAYOUT(layoutSignIn), buttonSignIn, 900, 450);  buttonSignUp = gtk\_button\_new\_with\_label("Buat Akun");  gtk\_layout\_put(GTK\_LAYOUT(layoutSignIn), buttonSignUp, 1070, 495);  // CSS  GdkDisplay \*display;  display = gdk\_display\_get\_default();  GdkScreen \*screen;  screen = gdk\_display\_get\_default\_screen(display);  GtkCssProvider \*css = gtk\_css\_provider\_new();  gtk\_css\_provider\_load\_from\_path(css, "src/css/sign\_in.css", NULL);  gtk\_style\_context\_add\_provider\_for\_screen(screen, GTK\_STYLE\_PROVIDER(css), GTK\_STYLE\_PROVIDER\_PRIORITY\_APPLICATION);  // DISPLAY  gtk\_widget\_show\_all(windowSignIn);  // SIGNAL  g\_signal\_connect(windowSignIn, "destroy", G\_CALLBACK(gtk\_main\_quit), NULL);  g\_signal\_connect(buttonSignUp, "clicked", G\_CALLBACK(handle\_display\_sign\_up), windowSignIn);  g\_signal\_connect(buttonSignIn, "clicked", G\_CALLBACK(check\_login), NULL);  gtk\_main();  }  void handle\_display\_sign\_in(GtkWidget \*widget, GtkWidget \*window){  gtk\_window\_close(GTK\_WINDOW(window));  display\_sign\_in();  } |

**sign\_up.c**

|  |
| --- |
| #include "sign\_up.h"  // WIDGET  GtkWidget \*windowSignUp;  // INPUT  GtkWidget \*entryUsernameSignUp;  GtkWidget \*entryPasswordSignUp;  GtkWidget \*entryRepeatPass;  bool search\_exist\_acc(char \*user, char \*hash){  UserInfo \*temp;  temp = Head;  while(temp != NULL){  if(strcmp(temp->Username, user) == 0 && strcmp(temp->Password, hash) == 0){  return true;  }  temp = temp->Next;  }  return false;  }  void create\_new\_acc(){  char pass[100], passRepeat[100], passHashing[100], username[100];  // RESET  reset\_string(pass, 100);  reset\_string(passRepeat, 100);  reset\_string(username, 100);  reset\_string(passHashing, 100);  const gchar \*gtextPass = gtk\_entry\_get\_text(GTK\_ENTRY(entryPasswordSignUp));  const gchar \*gtextPassRepeat = gtk\_entry\_get\_text(GTK\_ENTRY(entryRepeatPass));  const gchar \*gtextUsername = gtk\_entry\_get\_text(GTK\_ENTRY(entryUsernameSignUp));  sprintf(pass, "%s", gtextPass);  sprintf(passRepeat, "%s", gtextPassRepeat);  sprintf(username, "%s", gtextUsername);  strcpy(passHashing, pass);  strcpy(passHashing, do\_password\_hash(passHashing));  bool isFound;  isFound = search\_exist\_acc(username, passHashing);  if (strcmp(pass, "") == 0 || strcmp(username, "") == 0 ){  display\_error\_dialog("Username dan Password Harus Diisi!", windowSignUp);  }  else if(strcmp(pass, passRepeat) != 0){  display\_error\_dialog("Password Tidak Sama!", windowSignUp);  }  else if(isFound){  display\_error\_dialog("Akun Sudah Ada!", windowSignUp);  }  else{  // FILE OPERATION  FILE \*f;  f = fopen("data/user\_pass\_data.txt", "a");  fprintf(f,"%s|%s\n", username, passHashing);  fclose(f);  User = malloc(sizeof(UserInfo));  strcpy(User->Username, username);  strcpy(User->Password, passHashing);  User->Next = NULL;  if (Head == NULL){  Head = User;  }  else{  UserInfo \*prev;  prev = Head;  while(prev->Next != NULL){  prev = prev->Next;  }  prev->Next = User;  }  // RESET INPUT  gtk\_entry\_set\_text(GTK\_ENTRY(entryUsernameSignUp), "");  gtk\_entry\_set\_text(GTK\_ENTRY(entryPasswordSignUp), "");  gtk\_entry\_set\_text(GTK\_ENTRY(entryRepeatPass), "");  // DISPLAY SUCCESS  display\_info\_dialog("Akun Berhasi Dibuat!", windowSignUp);  }  // RESET  reset\_string(pass, 100);  reset\_string(passRepeat, 100);  reset\_string(username, 100);  reset\_string(passHashing, 100);  }  void display\_sign\_up(){  // WIDGET  GtkWidget \*layoutSignUp;  GtkWidget \*bgSignUp;  // LABEL  GtkWidget \*labelUsername;  GtkWidget \*labelPassword;  GtkWidget \*labelRepeatPass;  GtkWidget \*labelAlreadyAcc;  // BUTTON  GtkWidget \*buttonCreateNewAcc;  GtkWidget \*buttonBackToLogin;  // ICON  GdkPixbuf \*iconSignUp;  gtk\_init(NULL, NULL);  // WINDOW  windowSignUp = gtk\_window\_new(GTK\_WINDOW\_TOPLEVEL);  gtk\_window\_set\_title(GTK\_WINDOW(windowSignUp), "SIGN UP");  gtk\_window\_set\_default\_size(GTK\_WINDOW(windowSignUp), 1280, 720);  gtk\_window\_set\_position(GTK\_WINDOW(windowSignUp), GTK\_WIN\_POS\_CENTER);  // LAYOUT  layoutSignUp = gtk\_layout\_new(NULL, NULL);  gtk\_container\_add(GTK\_CONTAINER (windowSignUp), layoutSignUp);  // ICON  iconSignUp = gdk\_pixbuf\_new\_from\_file("src/image/icon.png", NULL);  gtk\_window\_set\_icon(GTK\_WINDOW(windowSignUp), iconSignUp);  // BACKGROUND  bgSignUp = gtk\_image\_new\_from\_file("src/image/new\_account.png");  gtk\_layout\_put(GTK\_LAYOUT(layoutSignUp), bgSignUp, 0, 0);  // LABEL  labelUsername = gtk\_label\_new("Username");  gtk\_layout\_put(GTK\_LAYOUT(layoutSignUp), labelUsername, 200, 172);  labelPassword = gtk\_label\_new("Password");  gtk\_layout\_put(GTK\_LAYOUT(layoutSignUp), labelPassword, 200, 258);  labelRepeatPass = gtk\_label\_new("Ulangi Password");  gtk\_layout\_put(GTK\_LAYOUT(layoutSignUp), labelRepeatPass, 200, 344);  // ENTRY  entryUsernameSignUp = gtk\_entry\_new();  gtk\_widget\_set\_size\_request(entryUsernameSignUp, 250, 40);  gtk\_layout\_put(GTK\_LAYOUT(layoutSignUp), entryUsernameSignUp, 200, 210);  entryPasswordSignUp = gtk\_entry\_new();  gtk\_widget\_set\_size\_request(entryPasswordSignUp, 250, 40);  gtk\_layout\_put(GTK\_LAYOUT(layoutSignUp), entryPasswordSignUp, 200, 290);  gtk\_entry\_set\_visibility(GTK\_ENTRY(entryPasswordSignUp), false); // password hidden  entryRepeatPass = gtk\_entry\_new();  gtk\_widget\_set\_size\_request(entryRepeatPass, 250, 40);  gtk\_layout\_put(GTK\_LAYOUT(layoutSignUp), entryRepeatPass, 200, 380);  gtk\_entry\_set\_visibility(GTK\_ENTRY(entryRepeatPass), false); // password hidden  labelAlreadyAcc = gtk\_label\_new("Sudah punya akun?");  gtk\_layout\_put(GTK\_LAYOUT(layoutSignUp), labelAlreadyAcc, 200, 500);  // BUTTON  buttonCreateNewAcc = gtk\_button\_new\_with\_label("Daftar");  gtk\_layout\_put(GTK\_LAYOUT(layoutSignUp), buttonCreateNewAcc, 200, 440);  buttonBackToLogin = gtk\_button\_new\_with\_label("Login");  gtk\_layout\_put(GTK\_LAYOUT(layoutSignUp), buttonBackToLogin, 375, 495);  // CSS  GdkDisplay \*display;  display = gdk\_display\_get\_default();  GdkScreen \*screen;  screen = gdk\_display\_get\_default\_screen(display);  GtkCssProvider \*css = gtk\_css\_provider\_new();  gtk\_css\_provider\_load\_from\_path(css, "src/css/sign\_up.css", NULL);  gtk\_style\_context\_add\_provider\_for\_screen(screen, GTK\_STYLE\_PROVIDER(css), GTK\_STYLE\_PROVIDER\_PRIORITY\_APPLICATION);  // DISPLAY  gtk\_widget\_show\_all(windowSignUp);  // SIGNAL  g\_signal\_connect(windowSignUp, "destroy", G\_CALLBACK(gtk\_main\_quit), NULL);  g\_signal\_connect(buttonBackToLogin, "clicked", G\_CALLBACK(handle\_display\_sign\_in), windowSignUp);  g\_signal\_connect(buttonCreateNewAcc, "clicked", G\_CALLBACK(create\_new\_acc), NULL);  gtk\_main();  }  void handle\_display\_sign\_up(GtkWidget \*widget, GtkWidget \*window){  gtk\_window\_close(GTK\_WINDOW(window));  display\_sign\_up();  } |

**dashboard.c**

|  |
| --- |
| #include "dashboard.h"  void open\_website(GtkWidget \*widget, GtkWidget \*window){  if (g\_strcmp0(gtk\_widget\_get\_name(widget), "trailer1") == 0){  gtk\_show\_uri\_on\_window(NULL, "https://www.youtube.com/watch?v=jtDRXPTZT-M", GDK\_CURRENT\_TIME, NULL);  }  else if (g\_strcmp0(gtk\_widget\_get\_name(widget), "trailer2") == 0){  gtk\_show\_uri\_on\_window(NULL, "https://www.youtube.com/watch?v=06pl59umDlw", GDK\_CURRENT\_TIME, NULL);  }  else if (g\_strcmp0(gtk\_widget\_get\_name(widget), "trailer3") == 0){  gtk\_show\_uri\_on\_window(NULL, "https://www.youtube.com/watch?v=cSp1dM2Vj48", GDK\_CURRENT\_TIME, NULL);  }  else if (g\_strcmp0(gtk\_widget\_get\_name(widget), "trailer4") == 0){  gtk\_show\_uri\_on\_window(NULL, "https://www.youtube.com/watch?v=ZjIquMdpsSA&t=5s", GDK\_CURRENT\_TIME, NULL);  }  }  void display\_dashboard(GtkWidget \*window){  gtk\_window\_close(GTK\_WINDOW(window));  // WIDGET  GtkWidget \*windowDashboard;  GtkWidget \*layoutDashboard;  GtkWidget \*bgDashboard;  // LABEL  GtkWidget \*labelTitle;  GtkWidget \*labelPlaying;  // BUTTON  GtkWidget \*buttonDashtoLogin;  GtkWidget \*buttonPembelian;  GtkWidget \*buttonPemilihanKursi;  GtkWidget \*buttonPembayaran;  GtkWidget \*buttonInformasi;  GtkWidget \*buttonTrailerFilm1;  GtkWidget \*buttonTrailerFilm2;  GtkWidget \*buttonTrailerFilm3;  GtkWidget \*buttonTrailerFilm4;  // ICON  GdkPixbuf \*iconDashboard;  gtk\_init(NULL, NULL);  // WINDOW  windowDashboard = gtk\_window\_new(GTK\_WINDOW\_TOPLEVEL);  gtk\_window\_set\_title(GTK\_WINDOW(windowDashboard), "Dashboard D'MOV1E");  gtk\_window\_set\_default\_size(GTK\_WINDOW(windowDashboard), 1280, 720);  gtk\_window\_set\_position(GTK\_WINDOW(windowDashboard), GTK\_WIN\_POS\_CENTER);  // LAYOUT  layoutDashboard = gtk\_layout\_new(NULL, NULL);  gtk\_container\_add(GTK\_CONTAINER (windowDashboard), layoutDashboard);  // ICON  iconDashboard = gdk\_pixbuf\_new\_from\_file("src/image/icon.png", NULL);  gtk\_window\_set\_icon(GTK\_WINDOW(windowDashboard), iconDashboard);  // BACKGROUND  bgDashboard = gtk\_image\_new\_from\_file("src/image/dashboard.png");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashboard), bgDashboard, 0, 0);  // LABEL  labelTitle = gtk\_label\_new("Dashboard");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashboard), labelTitle, 270, 50);  gtk\_widget\_set\_name(labelTitle, "labelTitle");  labelPlaying = gtk\_label\_new("Playing Now!");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashboard), labelPlaying, 880, 50);  gtk\_widget\_set\_name(labelPlaying, "labelPlaying");  // BUTTON  buttonDashtoLogin = gtk\_button\_new\_with\_label("Logout");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashboard), buttonDashtoLogin, 140, 565);  gtk\_widget\_set\_size\_request(buttonDashtoLogin, 400, 75);  buttonPembelian = gtk\_button\_new\_with\_label("Pembelian");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashboard), buttonPembelian, 140, 165);  gtk\_widget\_set\_size\_request(buttonPembelian, 400, 75);  buttonPemilihanKursi = gtk\_button\_new\_with\_label("Pemilihan Kursi");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashboard), buttonPemilihanKursi, 140, 265);  gtk\_widget\_set\_size\_request(buttonPemilihanKursi, 400, 75);  buttonPembayaran = gtk\_button\_new\_with\_label("Pembayaran");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashboard), buttonPembayaran, 140, 365);  gtk\_widget\_set\_size\_request(buttonPembayaran, 400, 75);  buttonInformasi = gtk\_button\_new\_with\_label("Informasi");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashboard), buttonInformasi, 140, 465);  gtk\_widget\_set\_size\_request(buttonInformasi, 400, 75);  // FILM POSTER  buttonTrailerFilm1 = gtk\_button\_new\_with\_label("");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashboard), buttonTrailerFilm1, 800, 170);  gtk\_widget\_set\_name(buttonTrailerFilm1, "trailer1");  gtk\_widget\_set\_size\_request(buttonTrailerFilm1, 141, 210);  buttonTrailerFilm2 = gtk\_button\_new\_with\_label("");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashboard), buttonTrailerFilm2, 980, 170);  gtk\_widget\_set\_name(buttonTrailerFilm2, "trailer2");  gtk\_widget\_set\_size\_request(buttonTrailerFilm2, 141, 210);  buttonTrailerFilm3 = gtk\_button\_new\_with\_label("");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashboard), buttonTrailerFilm3, 800, 425);  gtk\_widget\_set\_name(buttonTrailerFilm3, "trailer3");  gtk\_widget\_set\_size\_request(buttonTrailerFilm3, 141, 210);  buttonTrailerFilm4 = gtk\_button\_new\_with\_label("");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashboard), buttonTrailerFilm4, 980, 425);  gtk\_widget\_set\_name(buttonTrailerFilm4, "trailer4");  gtk\_widget\_set\_size\_request(buttonTrailerFilm4, 141, 210);  // CSS  GdkDisplay \*display;  display = gdk\_display\_get\_default();  GdkScreen \*screen;  screen = gdk\_display\_get\_default\_screen(display);  GtkCssProvider \*css = gtk\_css\_provider\_new();  gtk\_css\_provider\_load\_from\_path(css, "src/css/dashboard.css", NULL);  gtk\_style\_context\_add\_provider\_for\_screen(screen, GTK\_STYLE\_PROVIDER(css), GTK\_STYLE\_PROVIDER\_PRIORITY\_APPLICATION);  // DISPLAY  gtk\_widget\_show\_all(windowDashboard);  // SIGNAL  g\_signal\_connect(windowDashboard, "destroy", G\_CALLBACK(gtk\_main\_quit), NULL);  g\_signal\_connect(buttonDashtoLogin, "clicked", G\_CALLBACK(handle\_display\_sign\_in), windowDashboard);  g\_signal\_connect(buttonPembelian, "clicked", G\_CALLBACK(handle\_display\_dashboard\_pembelian), windowDashboard);  g\_signal\_connect(buttonPemilihanKursi, "clicked", G\_CALLBACK(handle\_display\_dashboard\_pemilihan\_kursi), windowDashboard);  g\_signal\_connect(buttonPembayaran, "clicked", G\_CALLBACK(handle\_display\_dashboard\_pembayaran), windowDashboard);  g\_signal\_connect(buttonInformasi, "clicked", G\_CALLBACK(handle\_display\_dashboard\_informasi), windowDashboard);  // TRAILER SIGNAL  g\_signal\_connect(buttonTrailerFilm1, "clicked", G\_CALLBACK(open\_website), windowDashboard);  g\_signal\_connect(buttonTrailerFilm2, "clicked", G\_CALLBACK(open\_website), windowDashboard);  g\_signal\_connect(buttonTrailerFilm3, "clicked", G\_CALLBACK(open\_website), windowDashboard);  g\_signal\_connect(buttonTrailerFilm4, "clicked", G\_CALLBACK(open\_website), windowDashboard);  gtk\_main();  }  void handle\_display\_dashboard(GtkWidget \*widget, GtkWidget \*window){  display\_dashboard(window);  } |

**dashboard\_informasi.c**

|  |
| --- |
| #include "dashboard\_informasi.h"  void display\_dashboard\_informasi(){  // WIDGET  GtkWidget \*windowDashInformasi;  GtkWidget \*layoutDashInformasi;  GtkWidget \*bgDashInformasi;  // BUTTON  GtkWidget \*buttonAbout;  GtkWidget \*buttonInfoQueue;  GtkWidget \*buttonHighTransaction;  GtkWidget \*buttonTransactionHistory;  GtkWidget \*buttonToDash;  GtkWidget \*labelTitle;  // ICON  GdkPixbuf \*iconDashInformasi;  gtk\_init(NULL, NULL);  // WINDOW  windowDashInformasi = gtk\_window\_new(GTK\_WINDOW\_TOPLEVEL);  gtk\_window\_set\_title(GTK\_WINDOW(windowDashInformasi), "Informasi");  gtk\_window\_set\_default\_size(GTK\_WINDOW(windowDashInformasi), 1280, 720);  gtk\_window\_set\_position(GTK\_WINDOW(windowDashInformasi), GTK\_WIN\_POS\_CENTER);  // LAYOUT  layoutDashInformasi = gtk\_layout\_new(NULL, NULL);  gtk\_container\_add(GTK\_CONTAINER (windowDashInformasi), layoutDashInformasi);  // ICON  iconDashInformasi = gdk\_pixbuf\_new\_from\_file("src/image/icon.png", NULL);  gtk\_window\_set\_icon(GTK\_WINDOW(windowDashInformasi), iconDashInformasi);  // BACKGROUND  bgDashInformasi = gtk\_image\_new\_from\_file("src/image/informasi.png");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashInformasi), bgDashInformasi, 0, 0);  // label  labelTitle = gtk\_label\_new("Informasi");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashInformasi), labelTitle, 570, 50);  gtk\_widget\_set\_name(labelTitle, "labelTitle");  // BUTTON  buttonToDash = gtk\_button\_new\_with\_label("Dashboard");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashInformasi), buttonToDash, 50,550);  gtk\_widget\_set\_size\_request(buttonToDash, 540, 70);  buttonInfoQueue = gtk\_button\_new\_with\_label("Info Antrian");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashInformasi), buttonInfoQueue, 50,150);  gtk\_widget\_set\_size\_request(buttonInfoQueue, 540, 70);  buttonHighTransaction = gtk\_button\_new\_with\_label("Pembeli Terbaik");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashInformasi), buttonHighTransaction, 50,350);  gtk\_widget\_set\_size\_request(buttonHighTransaction, 550, 70);  buttonAbout = gtk\_button\_new\_with\_label("Tentang Aplikasi");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashInformasi), buttonAbout, 50,450);  gtk\_widget\_set\_size\_request(buttonAbout, 540, 70);  buttonTransactionHistory = gtk\_button\_new\_with\_label("Riwayat Transaksi");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashInformasi), buttonTransactionHistory, 50,250);  gtk\_widget\_set\_size\_request(buttonTransactionHistory, 550, 70);  // CSS  GdkDisplay \*display;  display = gdk\_display\_get\_default();  GdkScreen \*screen;  screen = gdk\_display\_get\_default\_screen(display);  GtkCssProvider \*css = gtk\_css\_provider\_new();  gtk\_css\_provider\_load\_from\_path(css, "src/css/dashboard\_informasi.css", NULL);  gtk\_style\_context\_add\_provider\_for\_screen(screen, GTK\_STYLE\_PROVIDER(css), GTK\_STYLE\_PROVIDER\_PRIORITY\_APPLICATION);  // DISPLAY  gtk\_widget\_show\_all(windowDashInformasi);  // SIGNAL  g\_signal\_connect(windowDashInformasi, "destroy", G\_CALLBACK(gtk\_main\_quit), NULL);  g\_signal\_connect(buttonToDash, "clicked", G\_CALLBACK(handle\_display\_dashboard), windowDashInformasi);  g\_signal\_connect(buttonHighTransaction, "clicked", G\_CALLBACK(handle\_display\_informasi\_pembeli), windowDashInformasi);  g\_signal\_connect(buttonInfoQueue, "clicked", G\_CALLBACK(handle\_display\_informasi\_antrian), windowDashInformasi);  g\_signal\_connect(buttonAbout, "clicked", G\_CALLBACK(handle\_display\_informasi\_aplikasi), windowDashInformasi);  g\_signal\_connect(buttonTransactionHistory, "clicked", G\_CALLBACK(handle\_display\_informasi\_transaksi), windowDashInformasi);  gtk\_main();  }  void handle\_display\_dashboard\_informasi(GtkWidget \*widget, GtkWidget \*window){  gtk\_window\_close(GTK\_WINDOW(window));  display\_dashboard\_informasi();  } |

**dashboard\_informasi\_antrian.c**

|  |
| --- |
| #include "dashboard\_informasi\_antrian.h"  void display\_informasi\_antrian(){  // WIDGET  GtkWidget \*windowInfoAntrian;  GtkWidget \*layoutInfoAntrian;  GtkWidget \*bgInfoAntrian;  GtkWidget \*buttonToInformasi;  // LABEL  GtkWidget \*labelQueueSelect;  GtkWidget \*labelQueuePay;  GtkWidget \*labelTitle;  GtkWidget \*labelQueueName;  GtkWidget \*labelEmptyData;  // ICON  GdkPixbuf \*iconInfoAntrian;  gtk\_init(NULL, NULL);  // WINDOW  windowInfoAntrian = gtk\_window\_new(GTK\_WINDOW\_TOPLEVEL);  gtk\_window\_set\_title(GTK\_WINDOW(windowInfoAntrian), "Informasi Antrian");  gtk\_window\_set\_default\_size(GTK\_WINDOW(windowInfoAntrian), 1280, 720);  gtk\_window\_set\_position(GTK\_WINDOW(windowInfoAntrian), GTK\_WIN\_POS\_CENTER);  // LAYOUT  layoutInfoAntrian = gtk\_layout\_new(NULL, NULL);  gtk\_container\_add(GTK\_CONTAINER (windowInfoAntrian), layoutInfoAntrian);  // ICON  iconInfoAntrian = gdk\_pixbuf\_new\_from\_file("src/image/icon.png", NULL);  gtk\_window\_set\_icon(GTK\_WINDOW(windowInfoAntrian), iconInfoAntrian);  // BACKGROUND  bgInfoAntrian = gtk\_image\_new\_from\_file("src/image/informasi\_antrian.png");  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoAntrian), bgInfoAntrian, 0, 0);  // LABEL  labelTitle = gtk\_label\_new("Informasi Antrian");  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoAntrian), labelTitle, 500, 50);  gtk\_widget\_set\_name(labelTitle, "labelTitle");  // BUTTON  buttonToInformasi = gtk\_button\_new\_with\_label("Informasi");  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoAntrian), buttonToInformasi, 570,660);  labelQueueName = gtk\_label\_new("Antrian Pemilihan Kursi");  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoAntrian), labelQueueName, 300,200);  labelQueueName = gtk\_label\_new("Antrian Pembayaran Tiket");  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoAntrian), labelQueueName, 750,200);  labelEmptyData = gtk\_label\_new("Belum ada data");  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoAntrian), labelEmptyData, 300,250);  int x = 300, y = 250;  FILE \*f;  f = fopen("data/user\_ticket\_data.txt", "r");  int counterLine;  char ch, compUsr[100];  int counter, idx;  reset\_string(compUsr, 100);  counter = 0;  counterLine = 0;  idx = 1;  bool hasDeleted = false;  while(!feof(f) && counterLine < 10){  ch = fgetc(f);  if (ch == '\n'){  counterLine++;  labelQueueSelect = gtk\_label\_new(compUsr);  if (!hasDeleted){  gtk\_widget\_destroy(labelEmptyData);  hasDeleted = true;  }  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoAntrian), labelQueueSelect, x,y);  y += 50;  reset\_string(compUsr, 100);  counter = 0;  idx = 1;  }  else if (ch == '|'){  idx++;  counter = 0;  }  else{  if (idx == 1){  compUsr[counter] = ch;  counter++;  }  }  }  fclose(f);  labelEmptyData = gtk\_label\_new("Belum ada data");  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoAntrian), labelEmptyData, 750,250);  hasDeleted = false;  x = 750, y = 250;  f = fopen("data/user\_invoice\_data.txt", "r");  reset\_string(compUsr, 100);  counter = 0;  counterLine = 0;  idx = 1;  while(!feof(f) && counterLine < 10){  ch = fgetc(f);  if (ch == '\n'){  counterLine++;  labelQueuePay = gtk\_label\_new(compUsr);  if (!hasDeleted){  gtk\_widget\_destroy(labelEmptyData);  hasDeleted = true;  }  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoAntrian), labelQueuePay, x,y);  y += 50;  reset\_string(compUsr, 100);  counter = 0;  idx = 1;  }  else if (ch == '|'){  idx++;  counter = 0;  }  else{  if (idx == 1){  compUsr[counter] = ch;  counter++;  }  }  }  fclose(f);  // CSS  GdkDisplay \*display;  display = gdk\_display\_get\_default();  GdkScreen \*screen;  screen = gdk\_display\_get\_default\_screen(display);  GtkCssProvider \*css = gtk\_css\_provider\_new();  gtk\_css\_provider\_load\_from\_path(css, "src/css/dashboard\_informasi\_antrian.css", NULL);  gtk\_style\_context\_add\_provider\_for\_screen(screen, GTK\_STYLE\_PROVIDER(css), GTK\_STYLE\_PROVIDER\_PRIORITY\_APPLICATION);  // DISPLAY  gtk\_widget\_show\_all(windowInfoAntrian);  // SIGNAL  g\_signal\_connect(windowInfoAntrian, "destroy", G\_CALLBACK(gtk\_main\_quit), NULL);  g\_signal\_connect(buttonToInformasi, "clicked", G\_CALLBACK(handle\_display\_dashboard\_informasi), windowInfoAntrian);  gtk\_main();  }  void handle\_display\_informasi\_antrian(GtkWidget \*widget, GtkWidget \*window){  gtk\_window\_close(GTK\_WINDOW(window));  display\_informasi\_antrian();  } |

**dashboard\_informasi\_aplikasi.c**

|  |
| --- |
| #include "dashboard\_informasi\_aplikasi.h"  void display\_informasi\_aplikasi(){  GtkWidget \*windowInfoAplikasi;  GtkWidget \*layoutInfoAplikasi;  GtkWidget \*bgInfoAplikasi;  GtkWidget \*buttonToInformasi;  // ICON  GdkPixbuf \*iconInfoAplikasi;  gtk\_init(NULL, NULL);  // WINDOW  windowInfoAplikasi = gtk\_window\_new(GTK\_WINDOW\_TOPLEVEL);  gtk\_window\_set\_title(GTK\_WINDOW(windowInfoAplikasi), "Informasi Aplikasi");  gtk\_window\_set\_default\_size(GTK\_WINDOW(windowInfoAplikasi), 1280, 720);  gtk\_window\_set\_position(GTK\_WINDOW(windowInfoAplikasi), GTK\_WIN\_POS\_CENTER);  // LAYOUT  layoutInfoAplikasi = gtk\_layout\_new(NULL, NULL);  gtk\_container\_add(GTK\_CONTAINER (windowInfoAplikasi), layoutInfoAplikasi);  // ICON  iconInfoAplikasi = gdk\_pixbuf\_new\_from\_file("src/image/icon.png", NULL);  gtk\_window\_set\_icon(GTK\_WINDOW(windowInfoAplikasi), iconInfoAplikasi);  // BACKGROUND  bgInfoAplikasi = gtk\_image\_new\_from\_file("src/image/Tentang Aplikasi.png");  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoAplikasi), bgInfoAplikasi, 0, 0);  // BUTTON  buttonToInformasi = gtk\_button\_new\_with\_label("Informasi");  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoAplikasi), buttonToInformasi, 50,660);  // CSS  GdkDisplay \*display;  display = gdk\_display\_get\_default();  GdkScreen \*screen;  screen = gdk\_display\_get\_default\_screen(display);  GtkCssProvider \*css = gtk\_css\_provider\_new();  gtk\_css\_provider\_load\_from\_path(css, "src/css/dashboard\_informasi\_aplikasi.css", NULL);  gtk\_style\_context\_add\_provider\_for\_screen(screen, GTK\_STYLE\_PROVIDER(css), GTK\_STYLE\_PROVIDER\_PRIORITY\_APPLICATION);  // DISPLAY  gtk\_widget\_show\_all(windowInfoAplikasi);  // SIGNAL  g\_signal\_connect(windowInfoAplikasi, "destroy", G\_CALLBACK(gtk\_main\_quit), NULL);  g\_signal\_connect(buttonToInformasi, "clicked", G\_CALLBACK(handle\_display\_dashboard\_informasi), windowInfoAplikasi);  gtk\_main();  }  void handle\_display\_informasi\_aplikasi(GtkWidget \*widget, GtkWidget \*window){  gtk\_window\_close(GTK\_WINDOW(window));  display\_informasi\_aplikasi();  } |

**dashboard\_informasi\_transaksi.c**

|  |
| --- |
| #include "dashboard\_informasi\_aplikasi.h"  void display\_informasi\_aplikasi(){  GtkWidget \*windowInfoAplikasi;  GtkWidget \*layoutInfoAplikasi;  GtkWidget \*bgInfoAplikasi;  GtkWidget \*buttonToInformasi;  // ICON  GdkPixbuf \*iconInfoAplikasi;  gtk\_init(NULL, NULL);  // WINDOW  windowInfoAplikasi = gtk\_window\_new(GTK\_WINDOW\_TOPLEVEL);  gtk\_window\_set\_title(GTK\_WINDOW(windowInfoAplikasi), "Informasi Aplikasi");  gtk\_window\_set\_default\_size(GTK\_WINDOW(windowInfoAplikasi), 1280, 720);  gtk\_window\_set\_position(GTK\_WINDOW(windowInfoAplikasi), GTK\_WIN\_POS\_CENTER);  // LAYOUT  layoutInfoAplikasi = gtk\_layout\_new(NULL, NULL);  gtk\_container\_add(GTK\_CONTAINER (windowInfoAplikasi), layoutInfoAplikasi);  // ICON  iconInfoAplikasi = gdk\_pixbuf\_new\_from\_file("src/image/icon.png", NULL);  gtk\_window\_set\_icon(GTK\_WINDOW(windowInfoAplikasi), iconInfoAplikasi);  // BACKGROUND  bgInfoAplikasi = gtk\_image\_new\_from\_file("src/image/Tentang Aplikasi.png");  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoAplikasi), bgInfoAplikasi, 0, 0);  // BUTTON  buttonToInformasi = gtk\_button\_new\_with\_label("Informasi");  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoAplikasi), buttonToInformasi, 50,660);  // CSS  GdkDisplay \*display;  display = gdk\_display\_get\_default();  GdkScreen \*screen;  screen = gdk\_display\_get\_default\_screen(display);  GtkCssProvider \*css = gtk\_css\_provider\_new();  gtk\_css\_provider\_load\_from\_path(css, "src/css/dashboard\_informasi\_aplikasi.css", NULL);  gtk\_style\_context\_add\_provider\_for\_screen(screen, GTK\_STYLE\_PROVIDER(css), GTK\_STYLE\_PROVIDER\_PRIORITY\_APPLICATION);  // DISPLAY  gtk\_widget\_show\_all(windowInfoAplikasi);  // SIGNAL  g\_signal\_connect(windowInfoAplikasi, "destroy", G\_CALLBACK(gtk\_main\_quit), NULL);  g\_signal\_connect(buttonToInformasi, "clicked", G\_CALLBACK(handle\_display\_dashboard\_informasi), windowInfoAplikasi);  gtk\_main();  }  void handle\_display\_informasi\_aplikasi(GtkWidget \*widget, GtkWidget \*window){  gtk\_window\_close(GTK\_WINDOW(window));  display\_informasi\_aplikasi();  } |

**dashboard\_antrian\_pembeli.c**

|  |
| --- |
| #include "dashboard\_informasi\_pembeli.h"  int foundHighest[3];  char chrUsernameFoundHighest[3][100] = {"data kosong", "data kosong", "data kosong"};  char chrFilmNameFoundHighest[3][50] = {"data kosong", "data kosong", "data kosong"};  int counterFoundHigh = -1;  void find\_highest(BST \*curr){  if (curr != NULL){  find\_highest(curr->bRight);  counterFoundHigh++;  if (counterFoundHigh < 3){  strcpy(chrUsernameFoundHighest[counterFoundHigh], curr->bUsername);  strcpy(chrFilmNameFoundHighest[counterFoundHigh], curr->bFilmName);  foundHighest[counterFoundHigh] = curr->bPayment;  }  find\_highest(curr->bLeft);  }  }  void display\_informasi\_pembeli(){  bRoot = NULL;  // WIDGET  GtkWidget \*windowInfoBST;  GtkWidget \*layoutInfoBST;  GtkWidget \*bgInfoBST;  GtkWidget \*buttonToInformasi;  // LABEL  GtkWidget \*labelPembeliTerbaik;  GtkWidget \*labelTitle;  // ICON  GdkPixbuf \*iconInfoBST;  gtk\_init(NULL, NULL);  // WINDOW  windowInfoBST = gtk\_window\_new(GTK\_WINDOW\_TOPLEVEL);  gtk\_window\_set\_title(GTK\_WINDOW(windowInfoBST), "Informasi Pembeli Terbaik");  gtk\_window\_set\_default\_size(GTK\_WINDOW(windowInfoBST), 1280, 720);  gtk\_window\_set\_position(GTK\_WINDOW(windowInfoBST), GTK\_WIN\_POS\_CENTER);  // LAYOUT  layoutInfoBST = gtk\_layout\_new(NULL, NULL);  gtk\_container\_add(GTK\_CONTAINER (windowInfoBST), layoutInfoBST);  // ICON  iconInfoBST = gdk\_pixbuf\_new\_from\_file("src/image/icon.png", NULL);  gtk\_window\_set\_icon(GTK\_WINDOW(windowInfoBST), iconInfoBST);  // BACKGROUND  bgInfoBST = gtk\_image\_new\_from\_file("src/image/PEMBELI TERSULTAN.png");  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoBST), bgInfoBST, 0, 0);  // label  labelTitle = gtk\_label\_new("Pembeli Terbaik");  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoBST), labelTitle, 550, 50);  gtk\_widget\_set\_name(labelTitle, "labelTitle");  // BUTTON  buttonToInformasi = gtk\_button\_new\_with\_label("Informasi");  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoBST), buttonToInformasi, 600,650);  // load data from file  FILE \*f;  f = fopen("data/user\_transaction.txt", "r");  // make binary search tree from file  char compUsr[100], compFilmName[50];  int compPayment;  reset\_string(compUsr, 100);  reset\_string(compFilmName, 50);  compPayment = 0;  int counter = 0, idx = 1;  char ch;  while(!feof(f)){  ch = fgetc(f);  if (ch == '\n'){  // make tree  bNode = (BST\*)malloc(sizeof(BST));  strcpy(bNode->bUsername, compUsr);  strcpy(bNode->bFilmName, compFilmName);  bNode->bPayment = compPayment;  bNode->bLeft = NULL;  bNode->bRight = NULL;  if (bRoot == NULL){  bRoot = bNode;  }  else{  BST \*curr = bRoot, \*parent = NULL;  while(curr != NULL){  parent = curr;  if (bNode->bPayment > curr->bPayment){  curr = curr->bRight;  }  else{  curr = curr->bLeft;  }  }  if (bNode->bPayment > parent->bPayment){  parent->bRight = bNode;  }  else{  parent->bLeft = bNode;  }  }  // reset  reset\_string(compUsr, 100);  reset\_string(compFilmName, 50);  compPayment = 0;  counter = 0;  idx = 1;  }  else if (ch == '|'){  counter = 0;  idx++;  }  else{  if (idx == 1){  compUsr[counter] = ch;  counter++;  }  else if (idx == 2){  compFilmName[counter] = ch;  counter++;  }  else if (idx == 3){  int temp = ch - '0';  compPayment = (compPayment \* 10) + temp;  }  }  }  fclose(f);  // search top highest 3  if (bRoot != NULL){  find\_highest(bRoot);  }  // display it  int i, x, y;  for (i = 0; i < 3; i++){  if (i == 0){  x = 570;  y = 300;  }  else if (i == 1){  x = 300;  y = 430;  }  else{  x = 830;  y = 520;  }  labelPembeliTerbaik = gtk\_label\_new(chrUsernameFoundHighest[i]);  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoBST), labelPembeliTerbaik, x,y);  y += 20;  labelPembeliTerbaik = gtk\_label\_new(chrFilmNameFoundHighest[i]);  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoBST), labelPembeliTerbaik, x,y);  y += 20;  char temp[100];  reset\_string(temp, 100);  if (foundHighest[i] == 0){  strcpy(temp, "data kosong");  }  else{  sprintf(temp, "%d", foundHighest[i]);  }  labelPembeliTerbaik = gtk\_label\_new(temp);  gtk\_layout\_put(GTK\_LAYOUT(layoutInfoBST), labelPembeliTerbaik, x,y);  }  counterFoundHigh = -1;  // CSS  GdkDisplay \*display;  display = gdk\_display\_get\_default();  GdkScreen \*screen;  screen = gdk\_display\_get\_default\_screen(display);  GtkCssProvider \*css = gtk\_css\_provider\_new();  gtk\_css\_provider\_load\_from\_path(css, "src/css/dashboard\_informasi\_pembeli.css", NULL);  gtk\_style\_context\_add\_provider\_for\_screen(screen, GTK\_STYLE\_PROVIDER(css), GTK\_STYLE\_PROVIDER\_PRIORITY\_APPLICATION);  // DISPLAY  gtk\_widget\_show\_all(windowInfoBST);  // SIGNAL  g\_signal\_connect(windowInfoBST, "destroy", G\_CALLBACK(gtk\_main\_quit), NULL);  g\_signal\_connect(buttonToInformasi, "clicked", G\_CALLBACK(handle\_display\_dashboard\_informasi), windowInfoBST);  gtk\_main();  }  void handle\_display\_informasi\_pembeli(GtkWidget \*widget, GtkWidget \*window){  gtk\_window\_close(GTK\_WINDOW(window));  display\_informasi\_pembeli();  } |

**dashboard\_pembayaran.c**

|  |
| --- |
| #include "dashboard\_pembayaran.h"  // input  GtkWidget \*entryDibayarPelanggan;  GtkWidget \*layoutDashPembayaran;  GtkWidget \*labelPecahanUangTersedia;  GtkWidget \*labelPecahanUangKembali;  int arrUangKembalian[7];  void get\_user\_invoice\_from\_file(){  qFrontPay = NULL;  char compUsr[100], compBuyTime[50], compFilmName[50], compFilmTime[20], compFilmChair[50];  int compTicket, compStudioFilm, compStudioTime;  reset\_string(compUsr, 100);  reset\_string(compBuyTime, 50);  reset\_string(compFilmName, 50);  reset\_string(compFilmTime, 20);  reset\_string(compFilmChair, 50);  compTicket = 0;  compStudioFilm = 0;  compStudioTime = 0;  FILE \*f;  f = fopen("data/user\_invoice\_data.txt", "r");  char ch;  int idx = 1, counter = 0;  while(!feof(f)){  ch = fgetc(f);  if (ch == '|'){  counter = 0;  idx++;  }  else if (ch == '\n'){  qUser = malloc(sizeof(Queue));  strcpy(qUser->qUsername, compUsr);  qUser->qTiketDibeli = compTicket;  strcpy(qUser->qBuyTime, compBuyTime);  qUser->qStudioFilm = compStudioFilm;  qUser->qStudioTime = compStudioTime;  strcpy(qUser->qFilmName, compFilmName);  strcpy(qUser->qFilmTime, compFilmTime);  strcpy(qUser->qFilmChair, compFilmChair);  qUser->qNext = NULL;  if (qFrontPay == NULL){  qFrontPay = qUser;  }  else{  Queue \*prev;  prev = qFrontPay;  while(prev->qNext != NULL){  prev = prev->qNext;  }  prev->qNext = qUser;  }  // reset  reset\_string(compUsr, 100);  reset\_string(compBuyTime, 50);  reset\_string(compFilmName, 50);  reset\_string(compFilmTime, 20);  reset\_string(compFilmChair, 50);  compTicket = 0;  compStudioFilm = 0;  compStudioTime = 0;  counter = 0;  idx = 1;  }  else{  if (idx == 1){  compUsr[counter] = ch;  counter++;  }  else if(idx == 2){  int tempCompTicket = ch - '0';  compTicket = compTicket\*10 + tempCompTicket;  counter++;  }  else if (idx == 3){  compBuyTime[counter] = ch;  counter++;  }  else if (idx == 4){  int tempCompStudioFilm = ch - '0';  compStudioFilm = tempCompStudioFilm;  counter++;  }  else if (idx == 5){  int tempCompStudioTime = ch - '0';  compStudioTime = tempCompStudioTime;  counter++;  }  else if (idx == 6){  compFilmName[counter] = ch;  counter++;  }  else if (idx == 7){  compFilmTime[counter] = ch;  counter++;  }  else if (idx == 8){  compFilmChair[counter] = ch;  counter++;  }  }  }  fclose(f);  }  int get\_uang\_kembalian\_rec(int money, int depth){  if (money == 0){  return 0;  }  else if(money < MoneyCurrency[depth]){  get\_uang\_kembalian\_rec(money, depth+1);  }  else{  arrUangKembalian[depth] = money / MoneyCurrency[depth];  get\_uang\_kembalian\_rec(money % MoneyCurrency[depth], depth);  }  return 0;  }  void count\_cash\_back(GtkWidget \*widget, GtkWidget \*window){  char tempUangPelanggan[100];  reset\_string(tempUangPelanggan, 100);  const gchar \*gUangPelanggan = gtk\_entry\_get\_text(GTK\_ENTRY(entryDibayarPelanggan));  strcpy(tempUangPelanggan, gUangPelanggan);  // UANG FROM STRING TO INT  int i = 0, iUangPelanggan = 0;  while(tempUangPelanggan[i] != '\0'){  int temp = tempUangPelanggan[i] - '0';  iUangPelanggan = iUangPelanggan \* 10 + temp;  i++;  }  // GET TICKET PRICE  int saveIndex;  for (i = 0; i < 4; i++){  if (strcmp(qFrontPay->qFilmName, FilmName[i]) == 0){  saveIndex = i;  break;  }  }  if (iUangPelanggan < (qFrontPay->qTiketDibeli \* TicketPrice[saveIndex])){  display\_error\_dialog("Uang tidak cukup!", window);  }  else{  gtk\_widget\_destroy(labelPecahanUangTersedia);  gtk\_widget\_destroy(labelPecahanUangKembali);  // COUNT MONEY BACK  labelPecahanUangTersedia = gtk\_label\_new(  "Rp 100.000 : \n\n"  "Rp 50.000 : \n\n"  "Rp 20.000 : \n\n"  "Rp 10.000 : \n\n"  "Rp 5.000 : \n\n"  "Rp 2.000 : \n\n"  "Rp 1.000 : "  );  char tempPecahanUangKembali[50];  reset\_string(tempPecahanUangKembali, 50);  int uangKembalian = iUangPelanggan - (qFrontPay->qTiketDibeli \* TicketPrice[saveIndex]);  int i;  for (i = 0; i < 7; i++){  arrUangKembalian[i] = 0;  }  // RECURSIVE  get\_uang\_kembalian\_rec(uangKembalian, 0);  // DISPLAY MONEY BACK  for (i = 0; i < 7; i++){  char strTemp[100];  reset\_string(strTemp, 100);  sprintf(strTemp, "%d lembar\n\n", arrUangKembalian[i]);  strcat(tempPecahanUangKembali, strTemp);  }  labelPecahanUangKembali = gtk\_label\_new(tempPecahanUangKembali);  GtkWidget \*labelTitleKembalian;  labelTitleKembalian = gtk\_label\_new("Uang Kembalian");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelTitleKembalian, 950, 200);  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelPecahanUangTersedia, 950, 250);  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelPecahanUangKembali, 1065, 250);  gtk\_widget\_show\_all(window);  }  }  void display\_warn\_pembayaran(GtkWidget \*widget, GtkWidget \*window){  bool isDeleted = false;  if (qFrontPay == NULL){  display\_error\_dialog("Data pembayaran kosong", window);  }  else{  GtkWidget \*dialog;  dialog = gtk\_message\_dialog\_new(GTK\_WINDOW(window),  GTK\_DIALOG\_DESTROY\_WITH\_PARENT,  GTK\_MESSAGE\_ERROR,  GTK\_BUTTONS\_YES\_NO,  "Konfirmasi Pembayaran?");  gtk\_window\_set\_title(GTK\_WINDOW(dialog), "Warning");  int res = gtk\_dialog\_run(GTK\_DIALOG(dialog));  if (res == -8){ // YES  isDeleted = true;  // ADD TO TRANSACTION HISTORY  FILE \*f;  f = fopen("data/transaction\_history.txt", "a");  fprintf(f, "%s|%d|%s|%d|%d|%s|%s|%s\n",  qFrontPay->qUsername, qFrontPay->qTiketDibeli, qFrontPay->qBuyTime,  qFrontPay->qStudioFilm, qFrontPay->qStudioTime,  qFrontPay->qFilmName, qFrontPay->qFilmTime, qFrontPay->qFilmChair);  fclose(f);  // ADD TO USER TRANSACTION TO FIND BEST USER  f = fopen("data/user\_transaction.txt", "a");  int saveIndex, i;  for (i = 0; i < 4; i++){  if (strcmp(qFrontPay->qFilmName, FilmName[i]) == 0){  saveIndex = i;  break;  }  }  fprintf(f, "%s|%s|%d\n", qFrontPay->qUsername, qFrontPay->qFilmName, (qFrontPay->qTiketDibeli \* TicketPrice[saveIndex]));  fclose(f);  // DELETE IN QUEUE  if (qFrontPay->qNext == NULL){  free(qFrontPay);  qFrontPay = NULL;  }  else{  Queue \*curr = qFrontPay;  qFrontPay = qFrontPay->qNext;  free(curr);  curr = NULL;  }  // UPDATE DATA IN USER INVOICE  f = fopen("data/user\_invoice\_data.txt", "w");  Queue \*curr = qFrontPay;  while(curr != NULL){  fprintf(f, "%s|%d|%s|%d|%d|%s|%s|%s\n",  curr->qUsername, curr->qTiketDibeli, curr->qBuyTime,  curr->qStudioFilm, curr->qStudioTime,  curr->qFilmName, curr->qFilmTime, curr->qFilmChair);  curr = curr->qNext;  }  fclose(f);  // REMOVE ALL QUEUE  while(qFrontPay != NULL){  Queue \*curr = qFrontPay;  if (qFrontPay->qNext == NULL){  free(qFrontPay);  qFrontPay = NULL;  }  else{  qFrontPay = qFrontPay->qNext;  free(curr);  curr = NULL;  }  }  }  gtk\_widget\_destroy(dialog);  }  if(isDeleted){  // RELOAD TO READ DATA  gtk\_window\_close(GTK\_WINDOW(window));  display\_dashboard\_pembayaran();  }  }  void display\_dashboard\_pembayaran(){  get\_user\_invoice\_from\_file();  // WINDOW  GtkWidget \*windowDashPembayaran;  // ICON  GdkPixbuf \*iconDashPembayaran;  // BACKGROUND  GtkWidget \*bgDashPembayaran;  // BUTTON  GtkWidget \*buttonBackToDash;  GtkWidget \*buttonConfirmBayar;  GtkWidget \*buttonCountCashBack;  // LABEL  GtkWidget \*labelPembayaranNamaPembeli;  GtkWidget \*labelPembayaranJumlahTiket;  GtkWidget \*labelPembayaranWaktuPembelian;  GtkWidget \*labelPembayaranNamaFilm;  GtkWidget \*labelPembayaranStudioFilm;  GtkWidget \*labelPembayaranWaktuFilm;  GtkWidget \*labelPembayaranKursiDipilih;  GtkWidget \*labelPembayaranTotalHarga;  GtkWidget \*labelPembayaranDibayar;  GtkWidget \*labelUserPayInfo;  GtkWidget \*labelEmptyData;  GtkWidget \*labelTitle;  gtk\_init(NULL, NULL);  // WINDOW  windowDashPembayaran = gtk\_window\_new(GTK\_WINDOW\_TOPLEVEL);  gtk\_window\_set\_title(GTK\_WINDOW(windowDashPembayaran), "Pembayaran Tiket");  gtk\_window\_set\_default\_size(GTK\_WINDOW(windowDashPembayaran), 1280, 720);  gtk\_window\_set\_position(GTK\_WINDOW(windowDashPembayaran), GTK\_WIN\_POS\_CENTER);  // LAYOUT  layoutDashPembayaran = gtk\_layout\_new(NULL, NULL);  gtk\_container\_add(GTK\_CONTAINER (windowDashPembayaran), layoutDashPembayaran);  // ICON  iconDashPembayaran = gdk\_pixbuf\_new\_from\_file("src/image/icon.png", NULL);  gtk\_window\_set\_icon(GTK\_WINDOW(windowDashPembayaran), iconDashPembayaran);  // BACKGROUND  bgDashPembayaran = gtk\_image\_new\_from\_file("src/image/pembayaran.png");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), bgDashPembayaran, 0, 0);  // label  labelTitle = gtk\_label\_new("Pembayaran");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelTitle, 550, 50);  gtk\_widget\_set\_name(labelTitle, "labelTitle");  labelPecahanUangTersedia = gtk\_label\_new("");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelPecahanUangTersedia, 800, 100);  labelPecahanUangKembali = gtk\_label\_new("");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelPecahanUangKembali, 900, 100);  // label  labelPembayaranNamaPembeli = gtk\_label\_new("Nama: ");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelPembayaranNamaPembeli, 100, 200);  labelPembayaranJumlahTiket = gtk\_label\_new("Jumlah: ");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelPembayaranJumlahTiket, 100, 250);  labelPembayaranWaktuPembelian = gtk\_label\_new("Waktu beli: ");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelPembayaranWaktuPembelian, 100, 300);  labelPembayaranNamaFilm = gtk\_label\_new("Film: ");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelPembayaranNamaFilm, 100, 350);  labelPembayaranStudioFilm = gtk\_label\_new("Studio: ");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelPembayaranStudioFilm, 100, 400);  labelPembayaranWaktuFilm = gtk\_label\_new("Waktu: ");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelPembayaranWaktuFilm, 100, 450);  labelPembayaranKursiDipilih = gtk\_label\_new("Kursi: ");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelPembayaranKursiDipilih, 100, 500);  labelPembayaranTotalHarga = gtk\_label\_new("Total Harga: ");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelPembayaranTotalHarga, 500, 200);  labelPembayaranDibayar = gtk\_label\_new("Pembayaran: ");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelPembayaranDibayar, 500, 250);  // BUTTON  buttonBackToDash = gtk\_button\_new\_with\_label("Dashboard");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), buttonBackToDash, 150, 550);  buttonConfirmBayar = gtk\_button\_new\_with\_label("Konfirmasi Pembayaran");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), buttonConfirmBayar, 950, 550);  buttonCountCashBack = gtk\_button\_new\_with\_label("");  // DISPLAY IF THERE IS NO DATA  if (qFrontPay == NULL){  int i, y = 200;  for (i = 0; i < 7; i++){  labelEmptyData = gtk\_label\_new("Belum ada data");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelEmptyData, 200, y);  y += 50;  }  y = 200;  for (i = 0; i < 2; i++){  labelEmptyData = gtk\_label\_new("Belum ada data");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelEmptyData, 610, y);  y += 50;  }  }  // DISPLAY IF THERE IS DATA  if (qFrontPay != NULL){  gtk\_widget\_destroy(buttonCountCashBack);  buttonCountCashBack = gtk\_button\_new\_with\_label("Hitung Kembalian");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), buttonCountCashBack, 550, 550);  entryDibayarPelanggan = gtk\_entry\_new();  gtk\_widget\_set\_size\_request(entryDibayarPelanggan, 250, 40);  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), entryDibayarPelanggan, 500, 300);  // USER INFORMATION  labelUserPayInfo = gtk\_label\_new(qFrontPay->qUsername);  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelUserPayInfo, 200, 200);  char tempUserInfo[50];  reset\_string(tempUserInfo, 50);  sprintf(tempUserInfo, "%d", qFrontPay->qTiketDibeli);  labelUserPayInfo = gtk\_label\_new(tempUserInfo);  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelUserPayInfo, 200, 250);  char tempBuyTime[10];  reset\_string(tempBuyTime, 10);  int a = 0;  while(a < 9){  tempBuyTime[a] = qFrontPay->qBuyTime[a];  a++;  }  labelUserPayInfo = gtk\_label\_new(tempBuyTime);  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelUserPayInfo, 200, 300);  labelUserPayInfo = gtk\_label\_new(qFrontPay->qFilmName);  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelUserPayInfo, 200, 350);  reset\_string(tempUserInfo, 50);  sprintf(tempUserInfo, "%d", qFrontPay->qStudioFilm);  labelUserPayInfo = gtk\_label\_new(tempUserInfo);  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelUserPayInfo, 200, 400);  reset\_string(tempUserInfo, 50);  strcpy(tempUserInfo, StudioTime[(qFrontPay->qStudioFilm - 1)][(qFrontPay->qStudioTime - 1)]);  labelUserPayInfo = gtk\_label\_new(tempUserInfo);  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelUserPayInfo, 200, 450);  labelUserPayInfo = gtk\_label\_new(qFrontPay->qFilmChair);  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelUserPayInfo, 200, 500);  int i, saveIndex;  for (i = 0; i < 4; i++){  if (strcmp(qFrontPay->qFilmName, FilmName[i]) == 0){  saveIndex = i;  break;  }  }  reset\_string(tempUserInfo, 50);  sprintf(tempUserInfo, "%d", qFrontPay->qTiketDibeli \* TicketPrice[saveIndex]);  labelUserPayInfo = gtk\_label\_new(tempUserInfo);  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembayaran), labelUserPayInfo, 610, 200);  }  // CSS  GdkDisplay \*display;  display = gdk\_display\_get\_default();  GdkScreen \*screen;  screen = gdk\_display\_get\_default\_screen(display);  GtkCssProvider \*css = gtk\_css\_provider\_new();  gtk\_css\_provider\_load\_from\_path(css, "src/css/dashboard\_pembayaran.css", NULL);  gtk\_style\_context\_add\_provider\_for\_screen(screen, GTK\_STYLE\_PROVIDER(css), GTK\_STYLE\_PROVIDER\_PRIORITY\_APPLICATION);  // DISPLAY  gtk\_widget\_show\_all(windowDashPembayaran);  // SIGNAL  g\_signal\_connect(windowDashPembayaran, "destroy", G\_CALLBACK(gtk\_main\_quit), NULL);  g\_signal\_connect(buttonBackToDash, "clicked", G\_CALLBACK(handle\_display\_dashboard), windowDashPembayaran);  g\_signal\_connect(buttonCountCashBack, "clicked", G\_CALLBACK(count\_cash\_back), windowDashPembayaran);  g\_signal\_connect(buttonConfirmBayar, "clicked", G\_CALLBACK(display\_warn\_pembayaran), windowDashPembayaran);  gtk\_main();  }  void handle\_display\_dashboard\_pembayaran(GtkWidget \*widget, GtkWidget \*window){  gtk\_window\_close(GTK\_WINDOW(window));  display\_dashboard\_pembayaran();  } |

**dashboard\_pembelian.c**

|  |
| --- |
| #include "dashboard\_pembelian.h"  // WINDOW  GtkWidget \*windowDashPembelian;  // LAYOUT  GtkWidget \*layoutDashPembelian;  // LABEL  GtkWidget \*labelTimeNow;  // INPUT  GtkWidget \*entryNamaPembeli;  GtkWidget \*entryTiketDibeli;  // COMBOBOX  GtkWidget \*comboBox;  GtkWidget \*comboBoxWaktu;  bool exitDashPembelian;  void reset\_window(GtkWidget \*widget){  exitDashPembelian = true;  }  void combo\_selected(GtkWidget \*comboBox){  // GET ACTIVE TEXT FROM COMBOBOX  gchar \*getFilmDipilih = gtk\_combo\_box\_text\_get\_active\_text(GTK\_COMBO\_BOX\_TEXT(comboBox));  char chrGetFilmDipilih[50];  reset\_string(chrGetFilmDipilih, 50);  sprintf(chrGetFilmDipilih, "%s", getFilmDipilih);  // RESET COMBOBOX WAKTU  gtk\_combo\_box\_text\_remove\_all(GTK\_COMBO\_BOX\_TEXT(comboBoxWaktu));  // ADD COMBOBOX WAKTU WITH NEW DATA ACCORDING TO ACTIVE TEXT  int i,j;  for (i =0 ; i< 4; i++){  // FIND THE ACTIVE TEXT IN THE ARRAY  char compFilmDipilih[50];  reset\_string(compFilmDipilih, 50);  strcpy(compFilmDipilih, FilmName[i]);  if(strcmp(compFilmDipilih, chrGetFilmDipilih) == 0){  // IF FOUND  for (j =0; j < 3 ; j++){  // ADD COMBOBOX WAKTU WITH THE DATA  char appendText[20];  reset\_string(appendText, 20);  strcpy(appendText, FilmTime[i][j]);  gtk\_combo\_box\_text\_append\_text(GTK\_COMBO\_BOX\_TEXT(comboBoxWaktu), appendText);  }  break;  }  }  gtk\_widget\_show\_all(windowDashPembelian);  }  void display\_warn\_pembelian(GtkWidget \*widget, GtkWidget \*window){  GtkWidget \*dialog;  dialog = gtk\_message\_dialog\_new(GTK\_WINDOW(window),  GTK\_DIALOG\_DESTROY\_WITH\_PARENT,  GTK\_MESSAGE\_ERROR,  GTK\_BUTTONS\_YES\_NO,  "Konfirmasi Pembelian?");  gtk\_widget\_set\_name(dialog, "warnDialog");  gtk\_window\_set\_title(GTK\_WINDOW(dialog), "Warning");  int res = gtk\_dialog\_run(GTK\_DIALOG(dialog));  // GET USER INPUT IN GCHAR  const gchar \*gNamaPembeli = gtk\_entry\_get\_text(GTK\_ENTRY(entryNamaPembeli));  const gchar \*gTiketDibeli = gtk\_entry\_get\_text(GTK\_ENTRY(entryTiketDibeli));  if (res == -8){ // YES  char chrWaktuFilm[20];  reset\_string(chrWaktuFilm, 20);  // TIME CALC  time\_t rawtime;  struct tm \*timeInfo;  time(&rawtime);  timeInfo = localtime(&rawtime);  int timeNow, timeCompare = 0;  timeNow = (10000 \* timeInfo->tm\_hour) + (100\*timeInfo->tm\_min) + timeInfo->tm\_sec;  // GET TIME FILM  strcpy(chrWaktuFilm, gtk\_combo\_box\_text\_get\_active\_text(GTK\_COMBO\_BOX\_TEXT(comboBoxWaktu)));  // CONVERT TIME STRING TO INT FOR COMPARISON WITH TIME NOW  int i;  for (i = 0; i < 5; i++){  if (chrWaktuFilm[i] == '.'){  continue;  }  else{  int temp = chrWaktuFilm[i] - '0';  timeCompare = timeCompare \* 10 + temp;  }  }  timeCompare \*= 100;  // ERROR DETECTION  if (g\_strcmp0(gNamaPembeli, "") == 0 || g\_strcmp0(gTiketDibeli, "") == 0){  display\_error\_dialog("Nama atau tiket tidak boleh kosong!", window);  gtk\_widget\_destroy(dialog);  }  else if(g\_strcmp0(gTiketDibeli, "0") == 0){  display\_error\_dialog("Pembelian tiket minimal satu!", window);  gtk\_widget\_destroy(dialog);  }  else if(timeCompare < timeNow){  display\_error\_dialog("Waktu pembelian tiket telah ditutup!", window);  gtk\_widget\_destroy(dialog);  }  else{ // NO ERROR DETECTED  char chrTiketDibeli[5];  char chrFilmDipilih[50];  reset\_string(chrFilmDipilih, 50);  reset\_string(chrTiketDibeli, 5);  // CONVERT GCHAR TO CHAR  strcpy(chrFilmDipilih, gtk\_combo\_box\_text\_get\_active\_text(GTK\_COMBO\_BOX\_TEXT(comboBox)));  sprintf(chrTiketDibeli, "%s", gTiketDibeli);  // CONVERT STRING TIKET TO INT  int iTiketDibeli = 0, i = 0;  while(chrTiketDibeli[i] != '\0'){  int temp = chrTiketDibeli[i] - '0';  iTiketDibeli = iTiketDibeli \* 10 + temp;  i++;  }  /\* COUNT AVAILABLE CHAIR \*/  FILE \*f;  f = fopen("data/studio\_chair\_data.txt", "r");  char ch;  int counterLine = 0, counterStudio = 0, temp = 0;  while(!feof(f)){  ch = fgetc(f);  if (ch == '\n'){  AvailableChair[counterStudio][counterLine] = temp;  temp = 0;  counterLine++;  if (counterLine % 4 == 0){  counterLine = 0;  counterStudio++;  }  }  else{  if (ch == '0'){  temp++;  }  }  }  /\* END OF COUNT AVAILABLE CHAIR \*/  int tempStudioFilm, tempStdTime;  bool isDoraemon = false;  // COMPARISON  if (strcmp("KKN Desa Penari", chrFilmDipilih) == 0){  tempStudioFilm = 0;  }  else if (strcmp("The Throne", chrFilmDipilih) == 0){  tempStudioFilm = 1;  }  else if (strcmp("Ready Player One", chrFilmDipilih) == 0){  tempStudioFilm = 2;  }  else{  isDoraemon = true;  if (strcmp("09.00-10.30", chrWaktuFilm) == 0){  tempStudioFilm = 2;  tempStdTime = 0;  }  else if(strcmp("10.00-11.30", chrWaktuFilm) == 0){  tempStudioFilm = 1;  tempStdTime = 1;  }  else if(strcmp("12.00-13.30", chrWaktuFilm) == 0){  tempStudioFilm = 0;  tempStdTime = 2;  }  }  // DORAEMON PLAYS IN DIFFERENT STUDIO  if (!isDoraemon){  int i;  for (i = 0; i < 4; i++){  if (strcmp(chrWaktuFilm, StudioTime[tempStudioFilm][i]) == 0){  tempStdTime = i;  break;  }  }  }  // COMPARE TICKET WITH CHAIR AVAILABLE  if (AvailableChair[tempStudioFilm][tempStdTime] < iTiketDibeli){  display\_error\_dialog("Jumlah tiket melebihi batas!", window);  gtk\_widget\_destroy(dialog);  }  else{ // AVAILABLE  // CREATE NEW QUEUE  qUser = (Queue\*)malloc(sizeof(Queue));  // USERNAME DAN TIKET  strcpy(qUser->qUsername, gNamaPembeli);  qUser->qTiketDibeli = iTiketDibeli;  // TIME  time\_t rawtime;  struct tm \*timeInfo;  time(&rawtime);  timeInfo = localtime(&rawtime);  char chrTimeNow[100];  reset\_string(chrTimeNow, 100);  strcpy(chrTimeNow, timeToStr(timeInfo));  strcpy(qUser->qBuyTime, chrTimeNow);  // STUDIO FILM  bool isDoraemon = false;  if (strcmp("KKN Desa Penari", chrFilmDipilih) == 0){  qUser->qStudioFilm = 1;  }  else if (strcmp("The Throne", chrFilmDipilih) == 0){  qUser->qStudioFilm = 2;  }  else if (strcmp("Ready Player One", chrFilmDipilih) == 0){  qUser->qStudioFilm = 3;  }  else{  isDoraemon = true;  if (strcmp("09.00-10.30", chrWaktuFilm) == 0){  qUser->qStudioFilm = 3;  qUser->qStudioTime = 1;  }  else if(strcmp("10.00-11.30", chrWaktuFilm) == 0){  qUser->qStudioFilm = 2;  qUser->qStudioTime = 2;  }  else if(strcmp("12.00-13.30", chrWaktuFilm) == 0){  qUser->qStudioFilm = 1;  qUser->qStudioTime = 3;  }  }  // SEARCH FOR STUDIO TIME  int tempStudioTime = qUser->qStudioFilm;  tempStudioTime--;  // DORAEMON PLAYS IN DIFFERENT STUDIO  if (!isDoraemon){  int i;  for (i = 0; i < 4; i++){  if (strcmp(chrWaktuFilm, StudioTime[tempStudioTime][i]) == 0){  qUser->qStudioTime = (i+1);  break;  }  }  }  // COPY OTHER INFORMATION  strcpy(qUser->qFilmName, chrFilmDipilih);  strcpy(qUser->qFilmTime, chrWaktuFilm);  strcpy(qUser->qFilmChair, "");  // NEXT  qUser->qNext = NULL;  // ENQUEUE  if (qFront == NULL){  qFront = qUser;  }  else{  Queue \*prev = qFront;  while(prev->qNext != NULL){  prev = prev->qNext;  }  prev->qNext = qUser;  }  /\* FILE OPERATION \*/  FILE \*f;  f = fopen("data/user\_ticket\_data.txt", "a");  fprintf(f, "%s|%d|%s|%d|%d|%s|%s|%s\n",  qUser->qUsername, qUser->qTiketDibeli, qUser->qBuyTime,  qUser->qStudioFilm, qUser->qStudioTime,  qUser->qFilmName, qUser->qFilmTime, qUser->qFilmChair);  fclose(f);  /\* END OF FILE OPERATION \*/  // RESET INPUT  gtk\_entry\_set\_text(GTK\_ENTRY(entryNamaPembeli), "");  gtk\_entry\_set\_text(GTK\_ENTRY(entryTiketDibeli), "");  gtk\_widget\_destroy(dialog);  // SUCCESS  display\_info\_dialog("Pembelian berhasil", window);  }  }  }  else{  gtk\_widget\_destroy(dialog);  }  }  gboolean time\_handler(){  gchar buf[100];  if (exitDashPembelian){  return false;  }  GDateTime \*now = g\_date\_time\_new\_now\_local();  gchar \*my\_time = g\_date\_time\_format(now, "%H:%M:%S WITA");  sprintf(buf, "%s", my\_time);  gtk\_widget\_destroy(labelTimeNow);  labelTimeNow = gtk\_label\_new(buf);  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembelian), labelTimeNow, 1000, 100);  gtk\_widget\_show\_all(windowDashPembelian);  g\_free(my\_time);  g\_date\_time\_unref(now);  return true;  }  void display\_dashboard\_pembelian(){  exitDashPembelian = false;  // TIME CALC  time\_t rawtime;  struct tm \*timeInfo;  time(&rawtime);  timeInfo = localtime(&rawtime);  char chrTimeNow[50];  reset\_string(chrTimeNow, 20);  sprintf(chrTimeNow, "%.2d:%.2d.%.2d WITA", timeInfo->tm\_hour, timeInfo->tm\_min, timeInfo->tm\_sec);  int timeNow;  timeNow = (10000 \* timeInfo->tm\_hour) + (100\*timeInfo->tm\_min) + timeInfo->tm\_sec;  // TIME STATUS  char status[10];  reset\_string(status, 10);  // BACKGROUND  GtkWidget \*bgDashPembelian;  // LABEL  GtkWidget \*labelTitle;  GtkWidget \*labelNamaPembeli;  GtkWidget \*labelTiketDibeli;  GtkWidget \*labelFilmDipilih;  GtkWidget \*labelWaktuDipilih;  // BUTTON  GtkWidget \*buttonBackToDash;  GtkWidget \*buttonConfirmBeli;  GtkWidget \*buttonFilmTitle;  GtkWidget \*buttonFilmTime;  // ICON  GdkPixbuf \*iconDashPembelian;  gtk\_init(NULL, NULL);  // WINDOW  windowDashPembelian = gtk\_window\_new(GTK\_WINDOW\_TOPLEVEL);  gtk\_window\_set\_title(GTK\_WINDOW(windowDashPembelian), "PEMBELIAN TIKET");  gtk\_window\_set\_default\_size(GTK\_WINDOW(windowDashPembelian), 1280, 720);  gtk\_window\_set\_position(GTK\_WINDOW(windowDashPembelian), GTK\_WIN\_POS\_CENTER);  // LAYOUT  layoutDashPembelian = gtk\_layout\_new(NULL, NULL);  gtk\_container\_add(GTK\_CONTAINER (windowDashPembelian), layoutDashPembelian);  // ICON  iconDashPembelian = gdk\_pixbuf\_new\_from\_file("src/image/icon.png", NULL);  gtk\_window\_set\_icon(GTK\_WINDOW(windowDashPembelian), iconDashPembelian);  // BACKGROUND  bgDashPembelian = gtk\_image\_new\_from\_file("src/image/Pembelian.png");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembelian), bgDashPembelian, 0, 0);  // LABEL  labelTitle = gtk\_label\_new("Menu Pembelian");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembelian), labelTitle, 550, 75);  gtk\_widget\_set\_name(labelTitle, "labelTitle");  labelNamaPembeli = gtk\_label\_new("Nama");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembelian), labelNamaPembeli, 150, 150);  labelTiketDibeli = gtk\_label\_new("Jumlah Tiket:");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembelian), labelTiketDibeli, 150, 250);  labelTimeNow = gtk\_label\_new(chrTimeNow);  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembelian), labelTimeNow, 1000, 100);  // ENTRY  entryNamaPembeli = gtk\_entry\_new();  gtk\_widget\_set\_size\_request(entryNamaPembeli, 250, 40);  gtk\_entry\_set\_placeholder\_text(GTK\_ENTRY(entryNamaPembeli), "Masukkan nama pembeli...");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembelian), entryNamaPembeli, 150, 190);  entryTiketDibeli = gtk\_entry\_new();  gtk\_widget\_set\_size\_request(entryTiketDibeli, 250, 40);  gtk\_entry\_set\_placeholder\_text(GTK\_ENTRY(entryTiketDibeli), "Masukkan jumlah tiket...");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembelian), entryTiketDibeli, 150, 290);  // BUTTON  buttonBackToDash = gtk\_button\_new\_with\_label("Dashboard");  gtk\_widget\_set\_name(buttonBackToDash, "buttonAction");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembelian), buttonBackToDash, 150, 600);  buttonConfirmBeli = gtk\_button\_new\_with\_label("Buat Pesanan");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembelian), buttonConfirmBeli, 1000, 600);  gtk\_widget\_set\_name(buttonConfirmBeli, "buttonAction");  /\* FILM TIME DISPLAY \*/  int i,j;  gint x = 550, y = 200; // POSITION  for (i =0; i < 4; i++){  x = 550;  for (j = 0; j < 3; j++){  char filmTime[10];  reset\_string(filmTime, 10);  strcpy(filmTime, FilmTime[i][j]);  // MAKE NEW BUTTON  buttonFilmTime = gtk\_button\_new\_with\_label(filmTime);  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembelian), buttonFilmTime, x, y);  gtk\_widget\_set\_size\_request(buttonFilmTime, 35, 20);  // COMPARISON TIME  int k, compareTime = 0;  for (k = 0; k < 5; k++){  if (filmTime[k] == '.'){  continue;  }  else{  int temp = filmTime[k] - '0';  compareTime = compareTime \* 10 + temp;  }  }  compareTime \*= 100;  reset\_string(status, 10);  strcpy(status, "avail");  if (compareTime < timeNow){  reset\_string(status, 10);  strcpy(status, "late");  }  // GIVE COLOR TO BUTTON  gtk\_widget\_set\_name(buttonFilmTime, status);  x += 145;  }  y += 100;  }  /\* END OF FILM TIME DISPLAY \*/  /\* FILM NAME DISPLAY \*/  y = 150;  x = 550;  for (i = 0; i < 4; i++){  char filmName[50], chrTicketPrice[50];  reset\_string(filmName, 50);  reset\_string(chrTicketPrice, 50);  strcpy(filmName, FilmName[i]);  strcat(filmName, " - Rp. ");  sprintf(chrTicketPrice, "%d", TicketPrice[i]);  strcat(filmName, chrTicketPrice);  buttonFilmTitle = gtk\_button\_new\_with\_label(filmName);  gtk\_widget\_set\_name(buttonFilmTitle, "namaFilm");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembelian), buttonFilmTitle, x, y);  gtk\_widget\_set\_size\_request(buttonFilmTitle, 425, 50);  y += 100;  }  /\* END OF FILM NAME DISPLAY \*/  labelFilmDipilih = gtk\_label\_new("Pilih Film:");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembelian), labelFilmDipilih, 150, 350);  // COMBOBOX  comboBox = gtk\_combo\_box\_text\_new();  for (i =0 ; i< 4; i++){  char appendText[50];  reset\_string(appendText, 50);  strcpy(appendText, FilmName[i]);  gtk\_combo\_box\_text\_append\_text(GTK\_COMBO\_BOX\_TEXT(comboBox), appendText);  }  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembelian),comboBox, 150,390);  labelWaktuDipilih = gtk\_label\_new("Pilih Waktu:");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembelian), labelWaktuDipilih, 150, 440);  // COMBOBOX WAKTU  comboBoxWaktu = gtk\_combo\_box\_text\_new();  gtk\_combo\_box\_text\_append\_text(GTK\_COMBO\_BOX\_TEXT(comboBoxWaktu), "Silakan pilih film dahulu!");  gtk\_layout\_put(GTK\_LAYOUT(layoutDashPembelian),comboBoxWaktu, 150,480);  // CSS  GdkDisplay \*display;  display = gdk\_display\_get\_default();  GdkScreen \*screen;  screen = gdk\_display\_get\_default\_screen(display);  GtkCssProvider \*css = gtk\_css\_provider\_new();  gtk\_css\_provider\_load\_from\_path(css, "src/css/dashboard\_pembelian.css", NULL);  gtk\_style\_context\_add\_provider\_for\_screen(screen, GTK\_STYLE\_PROVIDER(css), GTK\_STYLE\_PROVIDER\_PRIORITY\_APPLICATION);  // DISPLAY  g\_timeout\_add(1000, (GSourceFunc)time\_handler, NULL);  gtk\_widget\_show\_all(windowDashPembelian);  // SIGNAL  g\_signal\_connect(windowDashPembelian, "destroy", G\_CALLBACK(gtk\_main\_quit), NULL);  g\_signal\_connect(windowDashPembelian, "destroy", G\_CALLBACK(reset\_window), NULL);  g\_signal\_connect(buttonBackToDash, "clicked", G\_CALLBACK(handle\_display\_dashboard), windowDashPembelian);  g\_signal\_connect(buttonConfirmBeli, "clicked", G\_CALLBACK(display\_warn\_pembelian), windowDashPembelian);  g\_signal\_connect(comboBox, "changed", G\_CALLBACK(combo\_selected), NULL);  gtk\_main();  }  void handle\_display\_dashboard\_pembelian(GtkWidget \*widget, GtkWidget \*window){  gtk\_window\_close(GTK\_WINDOW(window));  display\_dashboard\_pembelian();  } |

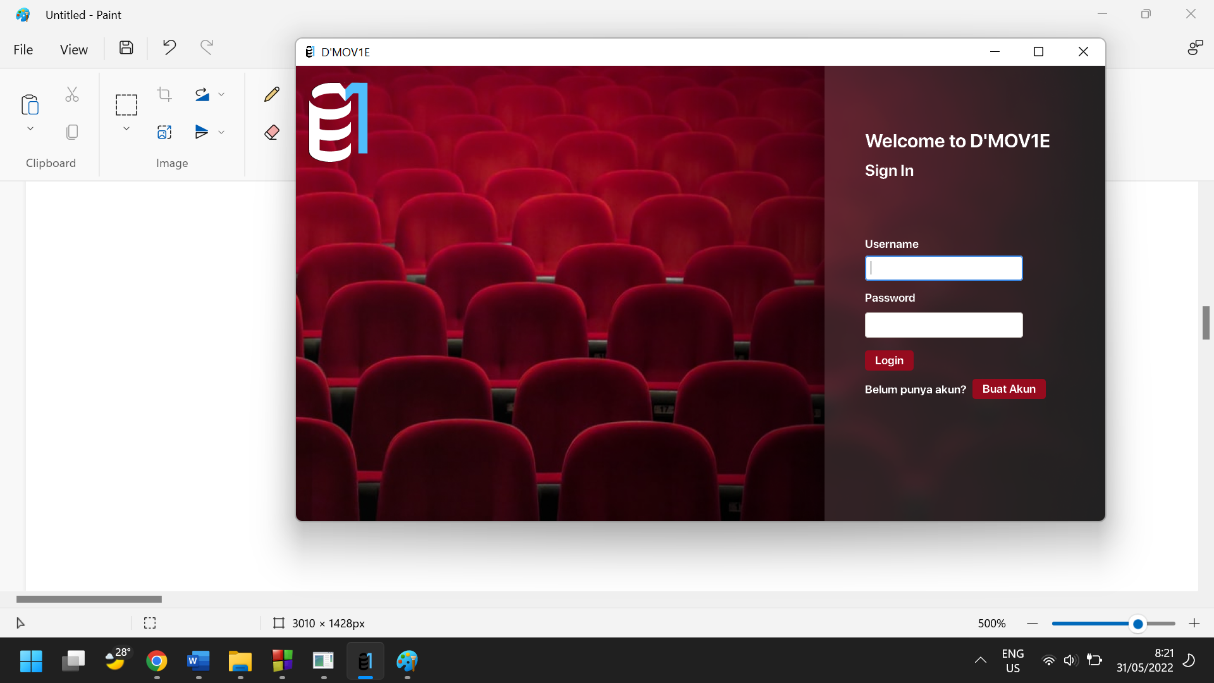
**dashboard\_pemilihan.c**

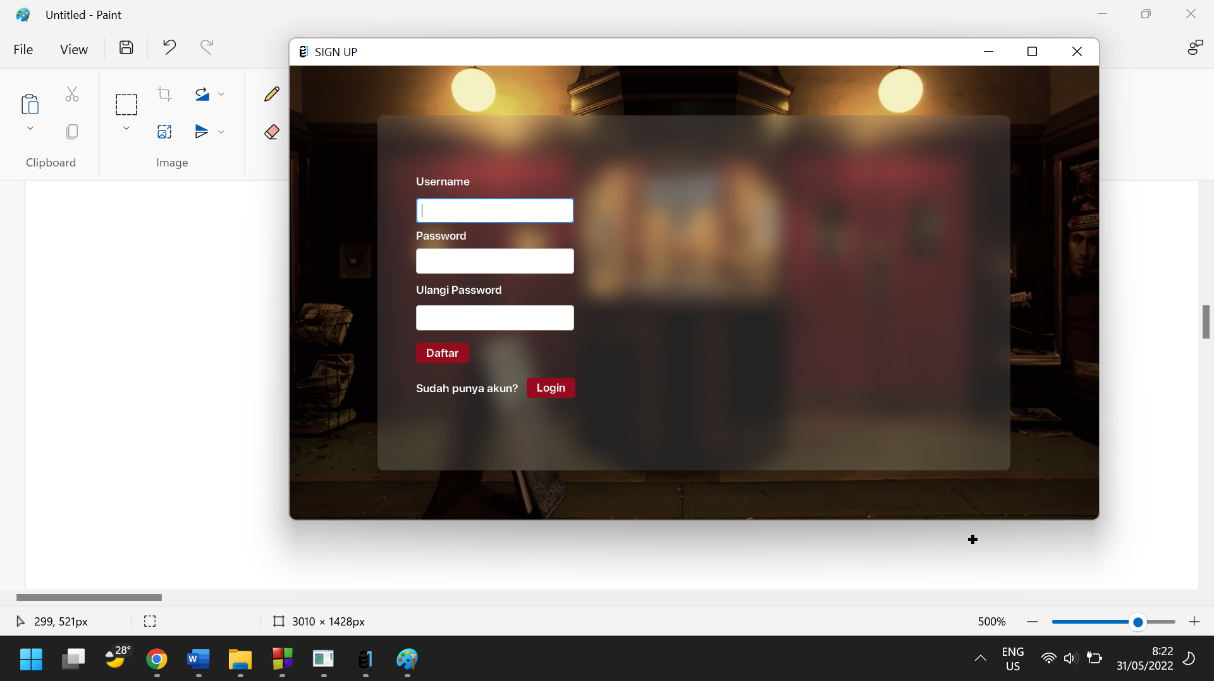
|  |
| --- |
| #include "dashboard\_pembelian.h"  // LAYOUT  GtkWidget \*layoutPemilihanKursi;  // LABEL  GtkWidget \*servePerson;  GtkWidget \*labelCountTicket;  GtkWidget \*labelStudioName;  GtkWidget \*labelFilmTime;  GtkWidget \*labelTiketDipilih;  GtkWidget \*labelEmptyData;  // BUTTON  GtkWidget \*buttonAddKursi;  GtkWidget \*buttonUndoKursi;  GtkWidget \*buttonKursi;  // COMBO BOX  GtkWidget \*comboSelectKursi;  int ChairSelected;  void get\_user\_ticket\_from\_file(){  char compUsr[100], compBuyTime[50], compFilmName[50], compFilmTime[20], compFilmChair[50];  int compTicket, compStudioFilm, compStudioTime;  reset\_string(compUsr, 100);  reset\_string(compBuyTime, 50);  reset\_string(compFilmName, 50);  reset\_string(compFilmTime, 20);  reset\_string(compFilmChair, 50);  compTicket = 0;  compStudioFilm = 0;  compStudioTime = 0;  qFront = NULL;  FILE \*f;  f = fopen("data/user\_ticket\_data.txt", "r");  char ch;  int idx = 1, counter = 0;  while(!feof(f)){  ch = fgetc(f);  if (ch == '|'){  counter = 0;  idx++;  }  else if (ch == '\n'){  qUser = malloc(sizeof(Queue));  strcpy(qUser->qUsername, compUsr);  qUser->qTiketDibeli = compTicket;  strcpy(qUser->qBuyTime, compBuyTime);  qUser->qStudioFilm = compStudioFilm;  qUser->qStudioTime = compStudioTime;  strcpy(qUser->qFilmName, compFilmName);  strcpy(qUser->qFilmTime, compFilmTime);  strcpy(qUser->qFilmChair, compFilmChair);  qUser->qNext = NULL;  if (qFront == NULL){  qFront = qUser;  }  else{  Queue \*prev;  prev = qFront;  while(prev->qNext != NULL){  prev = prev->qNext;  }  prev->qNext = qUser;  }  // reset  reset\_string(compUsr, 100);  reset\_string(compBuyTime, 50);  reset\_string(compFilmName, 50);  reset\_string(compFilmTime, 20);  reset\_string(compFilmChair, 50);  compTicket = 0;  compStudioFilm = 0;  compStudioTime = 0;  counter = 0;  idx = 1;  }  else{  if (idx == 1){  compUsr[counter] = ch;  counter++;  }  else if(idx == 2){  int tempCompTicket = ch - '0';  compTicket = compTicket\*10 + tempCompTicket;  counter++;  }  else if (idx == 3){  compBuyTime[counter] = ch;  counter++;  }  else if (idx == 4){  int tempCompStudioFilm = ch - '0';  compStudioFilm = tempCompStudioFilm;  counter++;  }  else if (idx == 5){  int tempCompStudioTime = ch - '0';  compStudioTime = tempCompStudioTime;  counter++;  }  else if (idx == 6){  compFilmName[counter] = ch;  counter++;  }  else if (idx == 7){  compFilmTime[counter] = ch;  counter++;  }  else if (idx == 8){  compFilmChair[counter] = ch;  counter++;  }  }  }  fclose(f);  }  void write\_user\_ticket\_to\_file(){  FILE \*f;  f = fopen("data/user\_ticket\_data.txt", "w");  Queue \*curr;  curr = qFront;  while(curr != NULL){  fprintf(f, "%s|%d|%s|%d|%d|%s|%s|%s\n",curr->qUsername, curr->qTiketDibeli, curr->qBuyTime, curr->qStudioFilm, curr->qStudioTime, curr->qFilmName, curr->qFilmTime, curr->qFilmChair);  curr = curr->qNext;  }  fclose(f);  }  void display\_warn\_pemilihan\_kursi(GtkWidget \*widget, GtkWidget \*window){  bool afterDel = false;  if (qFront == NULL){  display\_error\_dialog("Data pelanggan kosong", window);  }  else if(qFront->qTiketDibeli != ChairSelected){  display\_error\_dialog("Jumlah tiket tidak sama", window);  }  else{ // NO ERROR  afterDel = true;  GtkWidget \*dialog;  dialog = gtk\_message\_dialog\_new(GTK\_WINDOW(window),  GTK\_DIALOG\_DESTROY\_WITH\_PARENT,  GTK\_MESSAGE\_ERROR,  GTK\_BUTTONS\_YES\_NO,  "Konfirmasi Pemilihan Kursi?");  gtk\_widget\_set\_name(dialog, "warnConfirm");  gtk\_window\_set\_title(GTK\_WINDOW(dialog), "Warning");  int res = gtk\_dialog\_run(GTK\_DIALOG(dialog));  if (res == -8){ // YES  char tempWriteToData[100];  char tempFilmName[3];  reset\_string(tempWriteToData, 100);  reset\_string(tempFilmName, 3);  // GET DATA FROM TIKET DIPILIH  strcpy(tempWriteToData, gtk\_label\_get\_label(GTK\_LABEL(labelTiketDipilih)));  int i = 0, counter = 0;  while(tempWriteToData[i] != '\0'){  if (tempWriteToData[i] == '\n'){  reset\_string(tempFilmName, 3);  counter = 0;  }  else{  tempFilmName[counter] = tempWriteToData[i];  counter++;  if (counter == 2){  // SEARCH FILM CHAIR  int j, savePos;  for (j = 0; j < 16; j++){  if (strcmp(FilmChair[j], tempFilmName) == 0){  savePos = j;  j += 16; // BREAK LOOP  }  }  // CHANGE ARRAY  StudioChair[(qFront->qStudioFilm) - 1][(qFront->qStudioTime - 1)][savePos] = 1;  }  }  i++;  }  /\* FILE OPERATION STUDIO CHAIR DATA \*/  FILE \*f;  f = fopen("data/studio\_chair\_data.txt", "w");  int a, b, c;  for (a = 0; a < 3; a++){  for (b = 0; b < 4; b++){  for (c = 0; c < 16; c++){  fprintf(f, "%d", StudioChair[a][b][c]);  }  fprintf(f, "\n");  }  }  fclose(f);  /\* END OF FILE OPERATION STUDIO CHAIR DATA \*/  // COPY DATA TIKET TO QUEUE  char tempChairSelected[100];  reset\_string(tempChairSelected, 100);  Stack \*now = sTop;  while(now != NULL){  strcat(tempChairSelected, now->sChairName);  strcat(tempChairSelected, " ");  now = now->sNext;  }  strcpy(qFront->qFilmChair, tempChairSelected);  // WRITE TO FILE  f = fopen("data/user\_invoice\_data.txt", "a");  fprintf(f, "%s|%d|%s|%d|%d|%s|%s|%s\n",  qFront->qUsername, qFront->qTiketDibeli, qFront->qBuyTime,  qFront->qStudioFilm, qFront->qStudioTime,  qFront->qFilmName, qFront->qFilmTime, qFront->qFilmChair);  fclose(f);  // END OF WRITE TO FILE  // DEQUEUE  Queue \*curr;  curr = qFront;  while(curr->qNext != NULL){  curr = curr->qNext;  }  if(curr == qFront){  qFront = NULL;  free(curr);  curr = NULL;  }  else{  curr = qFront;  qFront = qFront->qNext;  free(curr);  curr = NULL;  }  // WRITE AGAIN  write\_user\_ticket\_to\_file();  gtk\_widget\_destroy(dialog);  display\_info\_dialog("Pemilihan Kursi Berhasil!", window);  }  else{  gtk\_widget\_destroy(dialog);  }  }  if(qFront == NULL){  gtk\_label\_set\_text(GTK\_LABEL(labelCountTicket), "Data kosong");  gtk\_label\_set\_text(GTK\_LABEL(servePerson), "Data kosong");  gtk\_label\_set\_text(GTK\_LABEL(labelStudioName), "Data kosong");  gtk\_label\_set\_text(GTK\_LABEL(labelFilmTime), "Data kosong");  gtk\_label\_set\_text(GTK\_LABEL(labelEmptyData), "Data kosong");  if (afterDel){  gtk\_window\_close(GTK\_WINDOW(window));  display\_dashboard\_pemilihan\_kursi();  }  }  else{  gtk\_widget\_destroy(labelTiketDipilih);  // REMOVE FROM STACK  while(sTop != NULL){  Stack \*curr = sTop;  sTop= sTop->sNext;  free(curr);  curr = NULL;  }  free(sTop);  sTop = NULL;  // CHANGE DISPLAY  char temp[3];  reset\_string(temp, 3);  sprintf(temp, "%d", qFront->qTiketDibeli);  gtk\_label\_set\_text(GTK\_LABEL(labelCountTicket), temp);  gtk\_label\_set\_text(GTK\_LABEL(servePerson), qFront->qUsername);  char tempStudioFilm[5];  reset\_string(tempStudioFilm, 5);  sprintf(tempStudioFilm, "%d", qFront->qStudioFilm);  gtk\_label\_set\_text(GTK\_LABEL(labelStudioName),tempStudioFilm);  gtk\_label\_set\_text(GTK\_LABEL(labelFilmTime), qFront->qFilmTime);  // RESET COMBO BOX  gtk\_combo\_box\_text\_remove\_all(GTK\_COMBO\_BOX\_TEXT(comboSelectKursi));  // ADD DATA TO COMBO BOX  int counter;  int i, j, delpos;  // COMBO BOX  for (i = 0; i < 16; i++){  char temp[5];  reset\_string(temp, 5);  strcpy(temp, FilmChair[i]);  gtk\_combo\_box\_text\_append\_text(GTK\_COMBO\_BOX\_TEXT(comboSelectKursi), temp);  }  // READ STUDIO CHAIR FROM FILE  FILE \*f;  f = fopen("data/studio\_chair\_data.txt", "r");  char ch;  int studio = 0, waktu = 0, kursi = 0;  while(!feof(f)){  ch = fgetc(f);  if (ch == '\n' && waktu == 3){  studio++;  waktu = 0;  kursi = 0;  }  else if (ch == '\n' && waktu < 3){  waktu++;  kursi = 0;  }  else{  int temp = ch - '0';  StudioChair[studio][waktu][kursi] = temp;  kursi++;  }  }  fclose(f);  // DELETE COMBO BOX  counter = 0;  delpos = counter;  gint x = 400, y = 260;  for (i = 0; i < 4; i++){  x = 400;  for (j = 0; j < 4; j++){  char copyCounter[3];  strcpy(copyCounter, FilmChair[counter]);  buttonKursi = gtk\_button\_new\_with\_label(copyCounter);  if (StudioChair[(qFront->qStudioFilm - 1)][(qFront->qStudioTime - 1)][counter] != 0){  gtk\_widget\_set\_name(buttonKursi, "late");  // DELETE FROM COMBO BOX  gtk\_combo\_box\_text\_remove(GTK\_COMBO\_BOX\_TEXT(comboSelectKursi), delpos);  }  else{  // add to stack  sChair = (Stack\*)malloc(sizeof(Stack));  strcpy(sChair->sChairName, FilmChair[counter]);  sChair->sNext = NULL;  if (sTopChair == NULL){  sTopChair = sChair;  }  else{  Stack \*prev = sTopChair;  while(prev->sNext != NULL){  prev = prev->sNext;  }  prev->sNext = sChair;  }  delpos++;  }  counter++;  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), buttonKursi, x, y);  gtk\_widget\_set\_size\_request(buttonKursi, 50,50);  x += 100;  }  y += 100;  }  gtk\_widget\_show\_all(window);  gtk\_window\_close(GTK\_WINDOW(window));  display\_dashboard\_pemilihan\_kursi();  }  }  void add\_kursi(GtkWidget \*widget, GtkWidget \*window){  if (sTopChair == NULL){  display\_error\_dialog("Kursi telah habis dipesan", window);  }  else{  // GET INPUT  char text[10];  reset\_string(text, 10);  strcpy(text, gtk\_combo\_box\_text\_get\_active\_text(GTK\_COMBO\_BOX\_TEXT(comboSelectKursi)));  // SEARCH POSITION TO REMOVE IN COMBOBOX  Stack \*curr;  curr = sTopChair;  int count = -1;  while(curr != NULL){  count++;  if (strcmp(curr->sChairName, text) == 0){  if (sTopChair->sNext == NULL){  sTopChair = NULL;  free(curr);  curr = NULL;  }  else{  if (curr == sTopChair){  sTopChair = sTopChair->sNext;  free(curr);  curr = NULL;  }  else{  Stack \*prev;  prev = sTopChair;  while(prev->sNext != curr){  prev = prev->sNext;  }  prev->sNext = curr->sNext;  curr->sNext = NULL;  free(curr);  curr = NULL;  }  }  break;  }  curr = curr->sNext;  }  gtk\_combo\_box\_text\_remove(GTK\_COMBO\_BOX\_TEXT(comboSelectKursi), count);  // ADD TO STACK  ChairSelected++;  sChair = (Stack\*)malloc(sizeof(Stack));  strcpy(sChair->sChairName, text);  sChair->sNext = NULL;  if (sTop == NULL){  sTop = sChair;  }  else{  Stack \*prev;  prev = sTop;  while(prev->sNext != NULL){  prev = prev->sNext;  }  prev->sNext = sChair;  }  // DISPLAY STACK  Stack \*now;  now = sTop;  char textDisplay[100];  reset\_string(textDisplay, 100);  while(now != NULL){  strcat(textDisplay, now->sChairName);  strcat(textDisplay, "\n");  now = now->sNext;  }  gtk\_widget\_destroy(GTK\_WIDGET(labelTiketDipilih));  labelTiketDipilih = gtk\_label\_new(textDisplay);  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), labelTiketDipilih, 800, 200);  gtk\_widget\_show\_all(window);  }  }  void undo\_kursi(GtkWidget \*widget, GtkWidget \*window){  char text[10];  reset\_string(text, 10);  ChairSelected--;  if (sTop == NULL){  display\_error\_dialog("Error belum ada dipilih!", window);  }  else{  // POP FROM STACK  Stack \*curr = sTop, \*prev = sTop;  if (sTop->sNext == NULL){  strcpy(text, sTop->sChairName);  free(sTop);  sTop = NULL;  }  else{  while(curr->sNext != NULL){  curr = curr->sNext;  }  while(prev->sNext != curr){  prev = prev->sNext;  }  prev->sNext = NULL;  strcpy(text, curr->sChairName);  free(curr);  curr = NULL;  }  // DISPLAY STACK  char textDisplay[100];  reset\_string(textDisplay, 100);  Stack \*now;  now = sTop;  while(now != NULL){  strcat(textDisplay, now->sChairName);  strcat(textDisplay, "\n");  now = now->sNext;  }  gtk\_widget\_destroy(GTK\_WIDGET(labelTiketDipilih));  labelTiketDipilih = gtk\_label\_new(textDisplay);  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), labelTiketDipilih, 800, 200);  gtk\_widget\_show\_all(window);  // ADD TO STACK CHAIR  sChair = (Stack\*)malloc(sizeof(Stack));  strcpy(sChair->sChairName, text);  sChair->sNext = NULL;  if(sTopChair == NULL){  sTopChair = sChair;  }  else{  Stack \*prev;  prev = sTopChair;  while(prev->sNext != NULL){  prev = prev->sNext;  }  prev->sNext = sChair;  }  // SORT STACK CHAIR  bool isSorted = false;  while(!isSorted){  isSorted = true;  Stack \*curr = sTopChair;  while (curr->sNext != NULL){  if (strcmp(curr->sChairName, curr->sNext->sChairName) > 0){  isSorted = false;  Stack \*temp;  temp = (Stack\*)malloc(sizeof(Stack));  strcpy(temp->sChairName, curr->sChairName);  strcpy(curr->sChairName, curr->sNext->sChairName);  strcpy(curr->sNext->sChairName, temp->sChairName);  free(temp);  temp = NULL;  }  curr = curr->sNext;  }  }  // SEARCH POSITION  int counter = -1;  curr = sTopChair;  while(curr != NULL){  counter++;  if(strcmp(curr->sChairName, text) == 0){  break;  }  curr = curr->sNext;  }  // ADD TO COMBOBOX AT RIGHT POSITION  gtk\_combo\_box\_text\_insert\_text(GTK\_COMBO\_BOX\_TEXT(comboSelectKursi), counter, text);  }  }  void display\_dashboard\_pemilihan\_kursi(){  get\_user\_ticket\_from\_file();  sTopChair = NULL;  sTop = NULL;  int counter = 0;  ChairSelected = 0;  GtkWidget \*windowPemilihanKursi;  GtkWidget \*bgPemilihanKursi;  GtkWidget \*labelTitle;  GtkWidget \*labelServePeople;  GtkWidget \*labelTicketAvail;  GtkWidget \*labelNoStudio;  GtkWidget \*labelTiketSementara;  // BUTTON  GtkWidget \*buttonBackToDash;  GtkWidget \*buttonConfirmPilih;  GtkWidget \*buttonScreenArea;  // ICON  GdkPixbuf \*iconPemilihanKursi;  gtk\_init(NULL, NULL);  // WINDOW  windowPemilihanKursi = gtk\_window\_new(GTK\_WINDOW\_TOPLEVEL);  gtk\_window\_set\_title(GTK\_WINDOW(windowPemilihanKursi), "Pemilihan Kursi");  gtk\_window\_set\_default\_size(GTK\_WINDOW(windowPemilihanKursi), 1280, 720);  gtk\_window\_set\_position(GTK\_WINDOW(windowPemilihanKursi), GTK\_WIN\_POS\_CENTER);  // LAYOUT  layoutPemilihanKursi = gtk\_layout\_new(NULL, NULL);  gtk\_container\_add(GTK\_CONTAINER (windowPemilihanKursi), layoutPemilihanKursi);  // ICON  iconPemilihanKursi= gdk\_pixbuf\_new\_from\_file("src/image/icon.png", NULL);  gtk\_window\_set\_icon(GTK\_WINDOW(windowPemilihanKursi), iconPemilihanKursi);  // BACKGROUND  bgPemilihanKursi = gtk\_image\_new\_from\_file("src/image/Pembelian.png");  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), bgPemilihanKursi, 0, 0);  labelTiketDipilih = gtk\_label\_new("");  labelTiketSementara = gtk\_label\_new("Tiket yang Sudah Dipilih:");  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), labelTiketSementara, 800, 150);  labelTitle = gtk\_label\_new("Menu Pemilihan Kursi");  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), labelTitle, 500, 75);  gtk\_widget\_set\_name(labelTitle, "labelTitle");  labelServePeople = gtk\_label\_new("Pelanggan:");  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), labelServePeople, 150, 200);  if(qFront == NULL){  labelEmptyData = gtk\_label\_new("Data kosong");  servePerson = gtk\_label\_new("Data kosong");  labelCountTicket = gtk\_label\_new("Data kosong");  labelStudioName = gtk\_label\_new("Data kosong");  labelFilmTime = gtk\_label\_new("Data kosong");  }  else{  labelEmptyData = gtk\_label\_new("");  char temp[3];  reset\_string(temp, 3);  sprintf(temp, "%d", qFront->qTiketDibeli);  labelCountTicket = gtk\_label\_new(temp);  servePerson = gtk\_label\_new(qFront->qUsername);  char tempStudioFilm[5];  reset\_string(tempStudioFilm, 5);  sprintf(tempStudioFilm, "%d", qFront->qStudioFilm);  labelStudioName = gtk\_label\_new(tempStudioFilm);  labelFilmTime = gtk\_label\_new(qFront->qFilmTime);  // BUTTON  buttonAddKursi = gtk\_button\_new\_with\_label("Tambahkan");  gtk\_widget\_set\_name(buttonAddKursi, "buttonAction");  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), buttonAddKursi, 150, 500);  buttonUndoKursi = gtk\_button\_new\_with\_label("Batalkan");  gtk\_widget\_set\_name(buttonUndoKursi, "buttonAction");  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), buttonUndoKursi, 300, 500);  buttonScreenArea = gtk\_button\_new\_with\_label("Screen Area");  gtk\_widget\_set\_name(buttonScreenArea, "screenArea");  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), buttonScreenArea, 500, 500);  gtk\_widget\_set\_size\_request(buttonScreenArea, 300, 10);  int i, j, delpos;  // COMBO BOX  comboSelectKursi = gtk\_combo\_box\_text\_new();  for (i = 0; i < 16; i++){  char temp[5];  reset\_string(temp, 5);  strcpy(temp, FilmChair[i]);  gtk\_combo\_box\_text\_append\_text(GTK\_COMBO\_BOX\_TEXT(comboSelectKursi), temp);  }  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), comboSelectKursi, 150, 400);  gtk\_widget\_set\_name(comboSelectKursi, "comboKursi");  gtk\_widget\_set\_size\_request(comboSelectKursi, 80, 30);  // READ DATA FROM STUDIO CHAIR DATA  FILE \*f;  f = fopen("data/studio\_chair\_data.txt", "r");  char ch;  int studio = 0, waktu = 0, kursi = 0;  while(!feof(f)){  ch = fgetc(f);  if (ch == '\n' && waktu == 3){  studio++;  waktu = 0;  kursi = 0;  }  else if (ch == '\n' && waktu < 3){  waktu++;  kursi = 0;  }  else{  int temp = ch - '0';  StudioChair[studio][waktu][kursi] = temp;  kursi++;  }  }  fclose(f);  // END OF READ DATA FROM STUDIO CHAIR DATA  // COMBOBOX CREATION  counter = 0;  delpos = counter;  gint x = 500, y = 210;  for (i = 0; i < 4; i++){  x = 500;  for (j = 0; j < 4; j++){  char copyCounter[3];  strcpy(copyCounter, FilmChair[counter]);  buttonKursi = gtk\_button\_new\_with\_label(copyCounter);  if (StudioChair[(qFront->qStudioFilm - 1)][(qFront->qStudioTime - 1)][counter] != 0){  gtk\_widget\_set\_name(buttonKursi, "late");  // DON'T ADD TO COMBOBOX  gtk\_combo\_box\_text\_remove(GTK\_COMBO\_BOX\_TEXT(comboSelectKursi), delpos);  }  else{  gtk\_widget\_set\_name(buttonKursi, "avail");  // ADD TO STACK CHAIR  sChair = (Stack\*)malloc(sizeof(Stack));  strcpy(sChair->sChairName, FilmChair[counter]);  sChair->sNext = NULL;  if (sTopChair == NULL){  sTopChair = sChair;  }  else{  Stack \*prev = sTopChair;  while(prev->sNext != NULL){  prev = prev->sNext;  }  prev->sNext = sChair;  }  delpos++;  }  counter++;  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), buttonKursi, x, y);  gtk\_widget\_set\_size\_request(buttonKursi, 50,50);  x += 70;  }  y += 70;  }  // ADD AND REMOVE CHAIR SIGNAL  g\_signal\_connect(buttonAddKursi, "clicked", G\_CALLBACK(add\_kursi), windowPemilihanKursi);  g\_signal\_connect(buttonUndoKursi, "clicked", G\_CALLBACK(undo\_kursi), windowPemilihanKursi);  }  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), servePerson,300, 200);  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), labelCountTicket, 300, 150);  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), labelStudioName, 230, 250);  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), labelFilmTime, 350, 250);  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), labelEmptyData, 800, 200);  // LABEL  labelTicketAvail = gtk\_label\_new("Jumlah Tiket: ");  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), labelTicketAvail, 150, 150);  labelNoStudio = gtk\_label\_new("Studio: ");  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), labelNoStudio, 150 ,250);  // BUTTON  buttonBackToDash = gtk\_button\_new\_with\_label("Dashboard");  gtk\_widget\_set\_name(buttonBackToDash, "buttonAction");  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi ), buttonBackToDash, 150, 600);  buttonConfirmPilih = gtk\_button\_new\_with\_label("Pilih Kursi");  gtk\_widget\_set\_name(buttonConfirmPilih, "buttonAction");  gtk\_layout\_put(GTK\_LAYOUT(layoutPemilihanKursi), buttonConfirmPilih, 1000, 600);  // CSS  GdkDisplay \*display;  display = gdk\_display\_get\_default();  GdkScreen \*screen;  screen = gdk\_display\_get\_default\_screen(display);  GtkCssProvider \*css = gtk\_css\_provider\_new();  gtk\_css\_provider\_load\_from\_path(css, "src/css/dashboard\_pemilihan.css", NULL);  gtk\_style\_context\_add\_provider\_for\_screen(screen, GTK\_STYLE\_PROVIDER(css), GTK\_STYLE\_PROVIDER\_PRIORITY\_APPLICATION);  // DISPLAY  gtk\_widget\_show\_all(windowPemilihanKursi);  // SIGNAL  g\_signal\_connect(windowPemilihanKursi, "destroy", G\_CALLBACK(gtk\_main\_quit), NULL);  g\_signal\_connect(buttonBackToDash, "clicked", G\_CALLBACK(handle\_display\_dashboard), windowPemilihanKursi);  g\_signal\_connect(buttonConfirmPilih, "clicked", G\_CALLBACK(display\_warn\_pemilihan\_kursi), windowPemilihanKursi);  gtk\_main();  }  void handle\_display\_dashboard\_pemilihan\_kursi(GtkWidget \*widget, GtkWidget \*window){  gtk\_window\_close(GTK\_WINDOW(window));  display\_dashboard\_pemilihan\_kursi();  } |

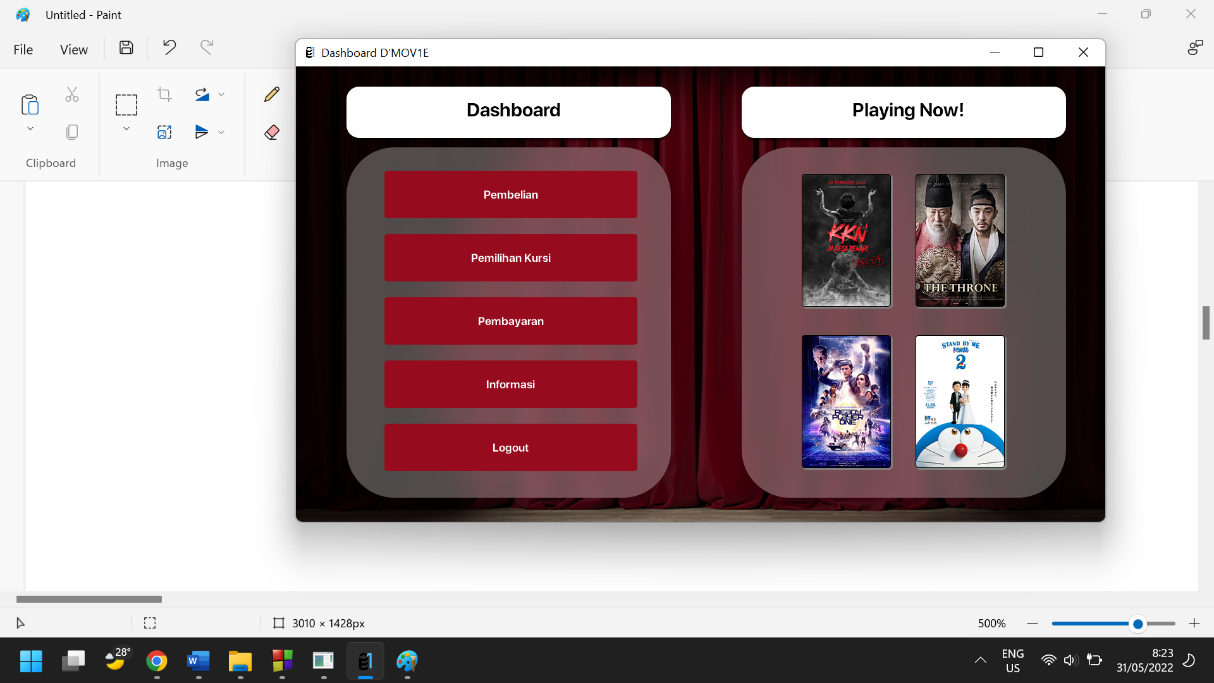
Informasi lengkap mengenai aplikasi dan source codenya dapat dilihat melalui tautan: <https://github.com/putuwaw/d-mov1e>

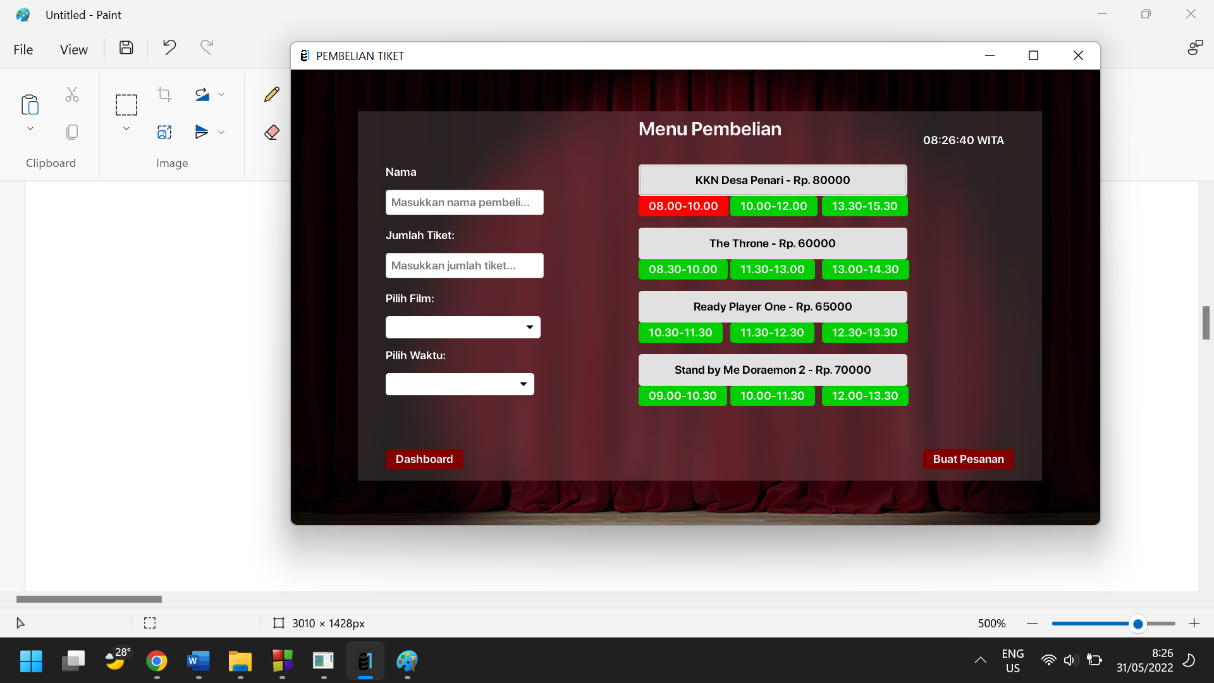
1. **Hasil Capture**

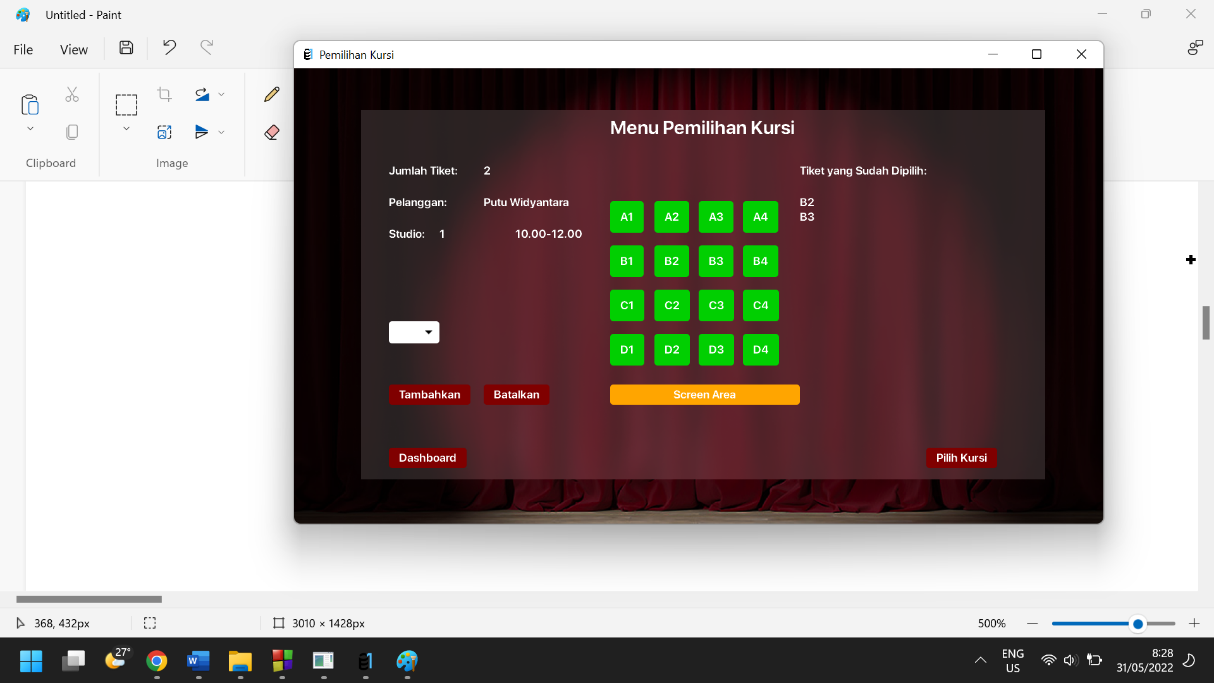
Berikut adalah hasil capture atau screenshot dari program dengan beberapa uji coba inputan:

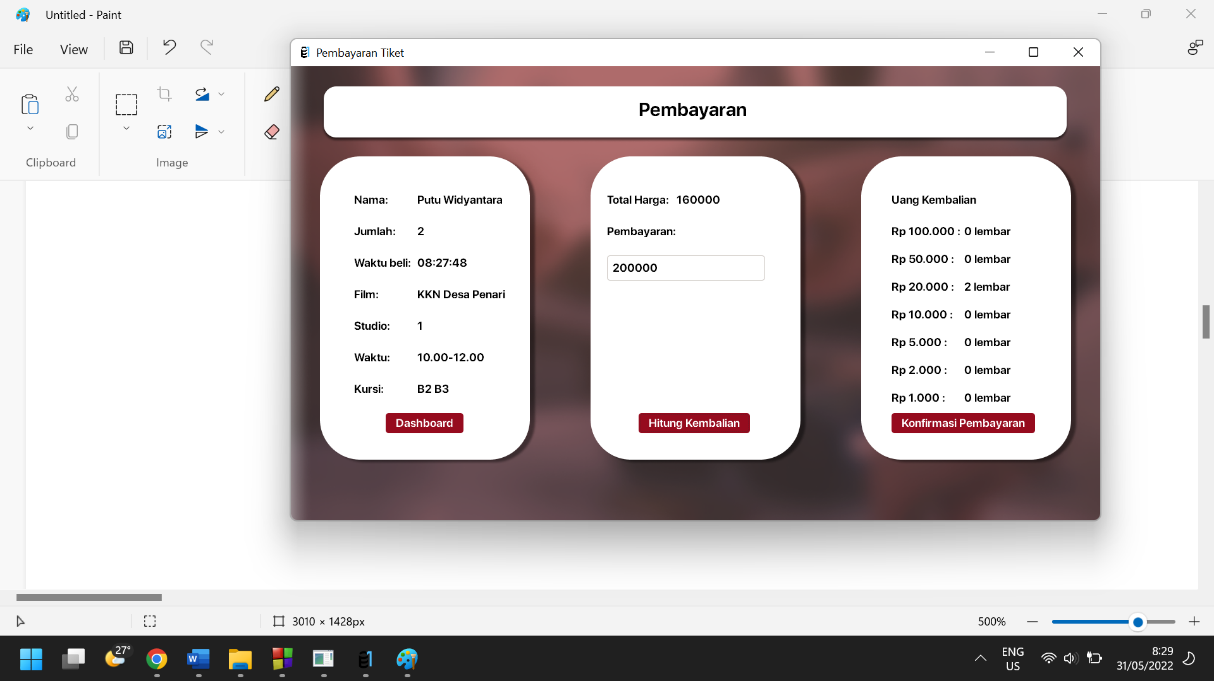


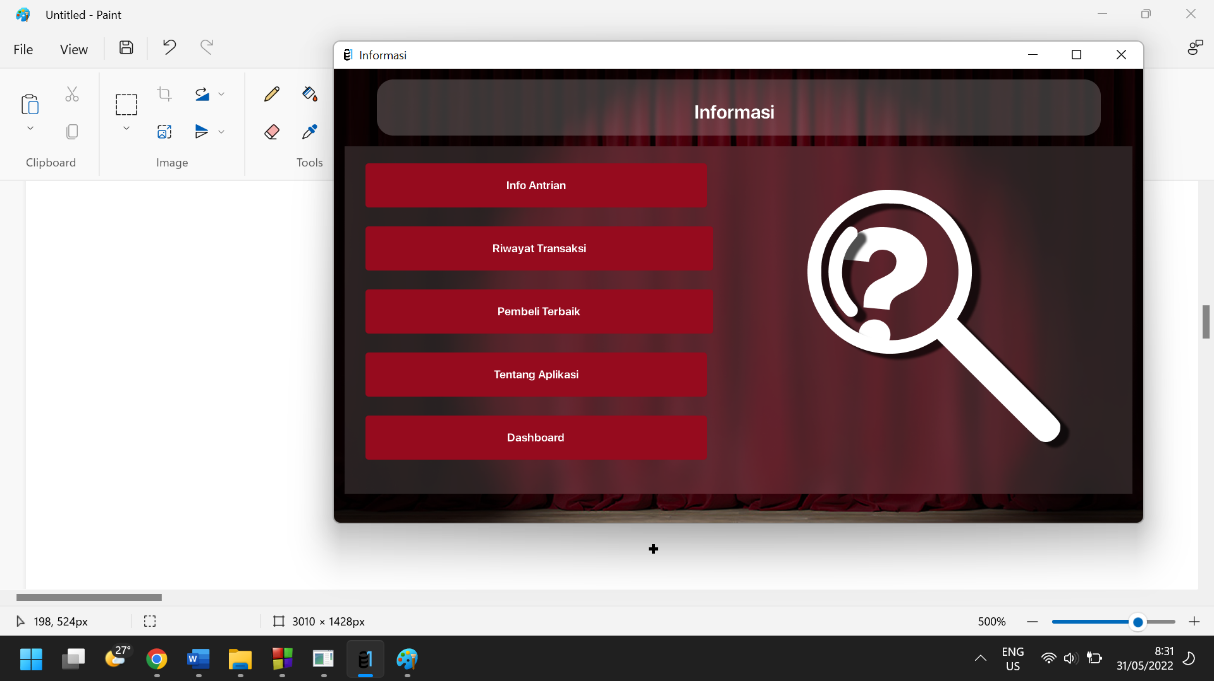


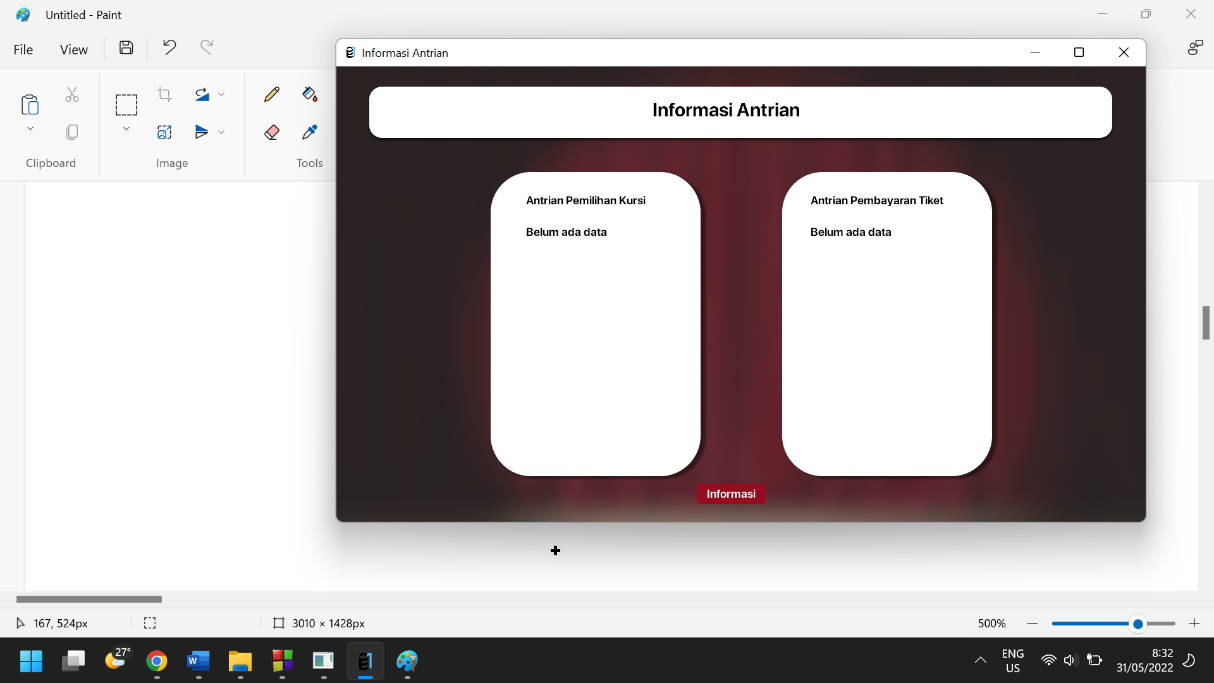


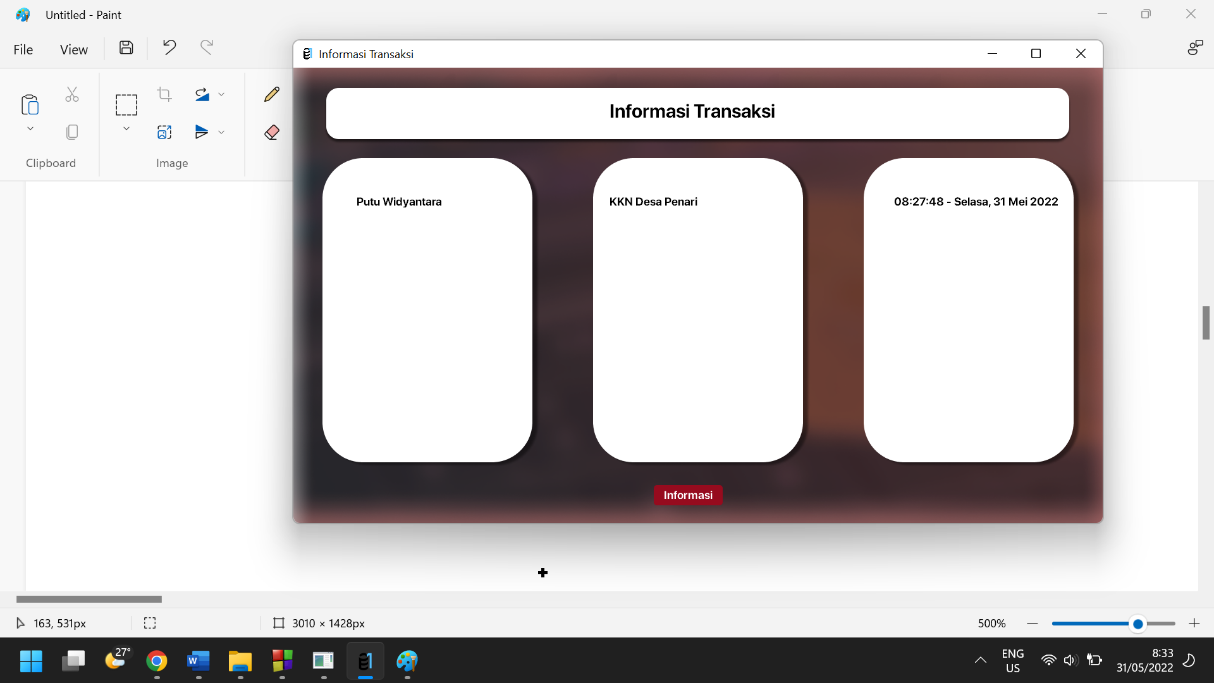


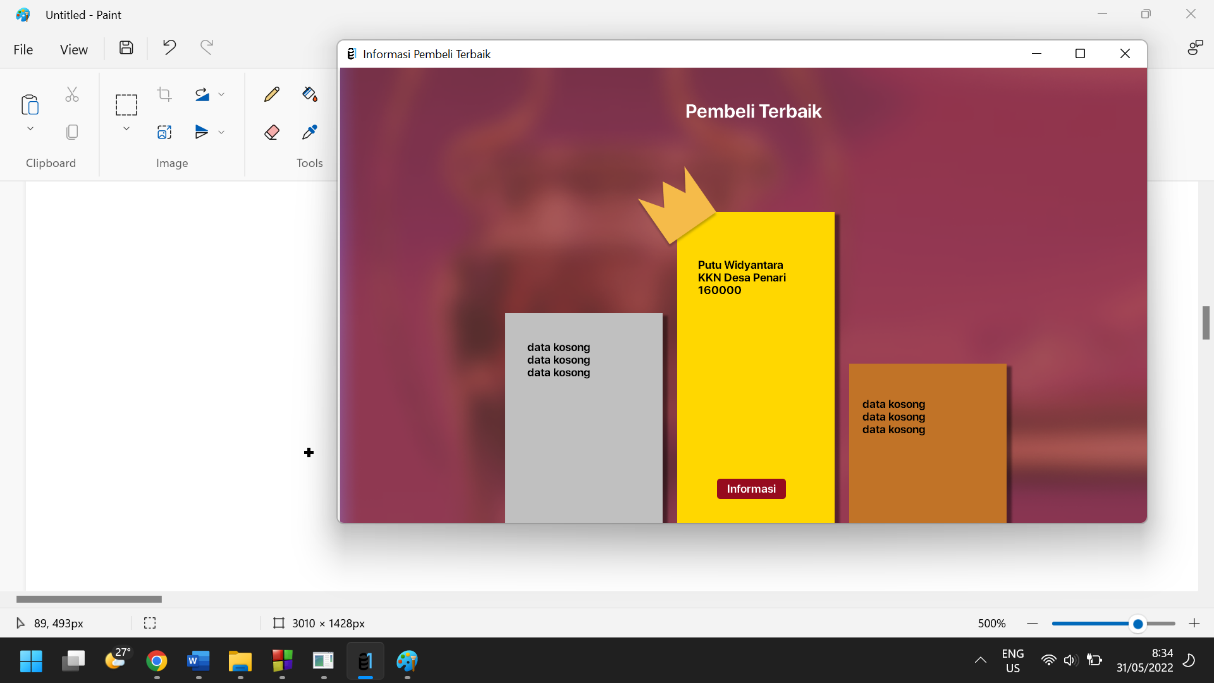














**BAB IV**

**PENUTUP**

1. **Kesimpulan**

Dari hasil praktikum tugas akhir yang sudah dikerjakan dan dibuat bersama-sama dengan anggota kelompok. Kami dapat mengambil kesimpulan bahwa pembuatan tugas akhir ini menggunakan hampir seluruh materi yang digunakan disaat perkuliahan dan contoh implementasinya seperti ketika membuat sign up page dimana kita memerlukan username dan password, tentunya password tersebut harus tetap menjadi rahasia sehingga menggunakan metode hashing untuk melakukan enkripsi pada password tersebut dan ketika membuat sign in page, kita diharuskan untuk membuat source code yang berfungsi untuk mengecek kecocokan masukan username dan password dari user dengan username dan password yang telah terdaftar di program tersebut atau pada database program. selain itu, queue juga berperan penting pada program ini yang digunakan untuk eksekusi pada user yang masuk terlebih dahulu jadi dengan algoritma “*first in first out*” ketika akan dilakukannya pemilihan kursi, sedangkan untuk pemilihan kursi ini menggunakan materi stack, dimana ketika akan melakukan penghapusan pilihan yang pertama kali dihapus adalah top of stack tersebut.