void Buyticket::openchangeticketsWindow(QString orderNumber, QString flightNumber, QString ExpensiveOrNot)

{

// 隐藏主窗口

//this->hide();

// 检查是否已经打开了改签窗口，防止重复创建

if (changeTicketsWindow != nullptr && changeTicketsWindow->isVisible()) {

changeTicketsWindow->raise();

changeTicketsWindow->activateWindow();

return;

}

// 查询原航班信息

QSqlQuery querypriFlight;

querypriFlight.prepare("SELECT flightstart, flightend "

"FROM flights "

"WHERE flightNumber = :flightNumber ");

// 绑定参数

querypriFlight.bindValue(":flightNumber", flightNumber);

querypriFlight.exec();

// 获取原航班的出发地和目的地

QString fromCity, toCity;

if (querypriFlight.next()) {

fromCity = querypriFlight.value(0).toString();

toCity = querypriFlight.value(1).toString();

} else {

qDebug() << "未找到原航班信息";

return;

}

// 查询原订单信息

QSqlQuery querypriOrder;

querypriOrder.prepare("SELECT flightDate "

"FROM orderlist "

"WHERE orderNumber = :orderNumber ");

// 绑定参数

querypriOrder.bindValue(":orderNumber", orderNumber);

querypriOrder.exec();

// 获取原航班的出发日期

QString flightDate;

if (querypriOrder.next()) {

flightDate = querypriOrder.value(0).toString();

} else {

qDebug() << "未找到原航班信息";

return;

}

// 创建改签窗口

changeTicketsWindow = new QWidget();

changeTicketsWindow->setFixedSize(900, 600); // 窗口大小固定为900x600

// 主布局

QVBoxLayout \*mainLayout = new QVBoxLayout();

// 窗口标题

QLabel \*titleLabel = new QLabel(QString("%1 -> %2 %3 当日其它航班信息")

.arg(fromCity) // 出发地

.arg(toCity) // 目的地

.arg(flightDate)); // 航班日期

titleLabel->setAlignment(Qt::AlignCenter);

titleLabel->setStyleSheet("font-size: 16px; color: #006F79;"); // 设置文本颜色为灰色

mainLayout->addWidget(titleLabel);

// 创建滚动区域

QScrollArea \*scrollArea = new QScrollArea();

scrollArea->setWidgetResizable(true); // 滚动区域内部大小可调整

// 滚动区域内容部件

QWidget \*scrollWidget = new QWidget();

QVBoxLayout \*ticketListLayout = new QVBoxLayout(scrollWidget); // 将滚动内容绑定到布局

// 清空之前的搜索结果

QLayoutItem\* item;

while ((item = ticketListLayout->takeAt(0)) != nullptr) {

delete item->widget();

delete item;

}

// 查询数据库并显示所有航班信息（同一出发地和目的地的其他航班）

QSqlQuery queryyyy;

queryyyy.prepare("SELECT flightNumber, flightStart, flightEnd, flightTimeStart, flightTimeEnd, "

"flightCheapMoney, flightExpensiveMoney, cheapOneRemaining, expensiveOneRemaining "

"FROM flights "

"WHERE flightStart = :fromCity AND flightEnd = :toCity "

"ORDER BY flightTimeStart ASC");

// 绑定参数

queryyyy.bindValue(":fromCity", fromCity);

queryyyy.bindValue(":toCity", toCity);

queryyyy.exec();

// 将查询结果展示到界面上

while (queryyyy.next()) {

QString newflightNumber = queryyyy.value(0).toString(); // 航班号

QString flightStart = queryyyy.value(1).toString(); // 出发地

QString flightEnd = queryyyy.value(2).toString(); // 目的地

QString flightTimeStart = queryyyy.value(3).toString().left(8); // 起飞时间

QString flightTimeEnd = queryyyy.value(4).toString().left(8); // 到达时间

QString flightCheapMoney = queryyyy.value(5).toString(); // 便宜舱位价格

QString flightExpensiveMoney = queryyyy.value(6).toString(); // 贵舱位价格

int cheapOneRemaining = queryyyy.value(7).toInt(); // 便宜舱位剩余数量

int expensiveOneRemaining = queryyyy.value(8).toInt(); // 贵舱位剩余数量

// 每个航班的容器

QFrame \*ticketFrame = new QFrame();

ticketFrame->setStyleSheet(R"(

QFrame {

background-color: #e0f7fa; /\* 浅蓝色背景 \*/

border: 1px solid #b2ebf2; /\* 浅蓝边框 \*/

border-radius: 7px; /\* 圆角 \*/

padding: 2px;

}

)");

ticketFrame->setFrameShape(QFrame::Box);

ticketFrame->setStyleSheet("border: 1px solid gray; border-radius: 4px;");

ticketFrame->setFixedHeight(65); // 未展开时高度

QVBoxLayout \*ticketLayout = new QVBoxLayout(ticketFrame); // 在 ticketFrame 内部创建布局

// 航班时间和航班号（可点击）

ClickableLabel \*headerLabel = new ClickableLabel();

headerLabel->setText(QString("%1 —— %2 | 航班号: %3")

.arg(flightTimeStart) // 起飞时间

.arg(flightTimeEnd) // 到达时间

.arg(newflightNumber)); // 航班号

headerLabel->setAlignment(Qt::AlignCenter);

headerLabel->setStyleSheet("font-size: 14px; font-weight: bold; color: #4b0082;"); // 设置字体大小、加粗和紫色

// 隐藏的详细信息框

QFrame \*detailFrame = new QFrame();

detailFrame->setFrameShape(QFrame::Box);

detailFrame->setStyleSheet(R"(

QFrame {

background-color: #e0f7fa; /\* 浅蓝色背景 \*/

border: 1px solid #b2ebf2; /\* 浅蓝边框 \*/

border-radius: 7px; /\* 圆角 \*/

padding: 2px;

}

QFrame[expanded="true"] {

background-color: #B0E0E6; /\* 展开时的浅灰色背景--------------------------- \*/

border-width: 0px 0px 0px 0px; /\* 展开时边框变窄 \*/

border-color: #D8B7DD; /\* 浅蓝色边框 \*/

}

)");

detailFrame->setStyleSheet("border: 1px solid gray; border-radius: 5px;");

detailFrame->setVisible(false); // 默认隐藏

detailFrame->setFixedHeight(100); // 设置展开后的高度

// 设置一个属性来标识是否展开

detailFrame->setProperty("expanded", false); // 默认未展开

// 创建展开和收起的函数

auto toggleDetailFrame = [detailFrame]() {

bool expanded = detailFrame->property("expanded").toBool();

if (expanded) {

detailFrame->setProperty("expanded", false);

detailFrame->setFixedHeight(65); // 收起时恢复高度

} else {

detailFrame->setProperty("expanded", true);

detailFrame->setFixedHeight(100); // 展开时高度增加

}

detailFrame->style()->unpolish(detailFrame); // 重新应用样式

detailFrame->style()->polish(detailFrame);

detailFrame->update(); // 更新界面

};

// 绑定事件（如点击headerLabel时展开或收起）

headerLabel->setClickCallback(toggleDetailFrame);

QVBoxLayout \*detailLayout = new QVBoxLayout();

// 头等舱信息

QHBoxLayout \*firstClassLayout = new QHBoxLayout();

QLabel \*firstClassLabel = new QLabel(QString("头等舱: ￥%1/人 (剩余%2张)").arg(flightExpensiveMoney).arg(expensiveOneRemaining));

firstClassLabel->setStyleSheet("font-size: 12px; color: #4b0082;");

firstClassLabel->setStyleSheet(R"(

QLabel {

background-color: #7EC8E3; /\* 设置背景色为 #7EC8E3 \*/

color: white; /\* 设置字体颜色为白色 \*/

font-size: 12px; /\* 字体大小 \*/

padding: 5px; /\* 内边距 \*/

border-radius: 5px; /\* 圆角 \*/

}

)");

QPushButton \*firstClassButton = new QPushButton("改签为");

firstClassButton->setStyleSheet(R"(

QPushButton {

background-color: #219ebc; /\* 浅蓝色背景 \*/

color: white;

padding: 5px;

border-radius: 5px;

font-size: 12px;

}

QPushButton:hover {

background-color: #8ecae6; /\* 悬停时变深 \*/

}

)");

firstClassButton->setFixedWidth(80);

firstClassLayout->addWidget(firstClassLabel);

firstClassLayout->addWidget(firstClassButton);

detailLayout->addLayout(firstClassLayout);

// 经济舱信息

QHBoxLayout \*economyClassLayout = new QHBoxLayout();

QLabel \*economyClassLabel = new QLabel(QString("经济舱: ￥%1/人 (剩余%2张)").arg(flightCheapMoney).arg(cheapOneRemaining));

economyClassLabel->setStyleSheet("font-size: 12px; color: #4b0082;");

economyClassLabel->setStyleSheet(R"(

QLabel {

background-color: #7EC8E3; /\* 设置背景色为 #7EC8E3 \*/

color: white; /\* 设置字体颜色为白色 \*/

font-size: 12px; /\* 字体大小 \*/

padding: 5px; /\* 内边距 \*/

border-radius: 5px; /\* 圆角 \*/

}

)");

QPushButton \*economyClassButton = new QPushButton("改签为");

economyClassButton->setStyleSheet(R"(

QPushButton {

background-color: #219ebc; /\* 浅蓝色背景 \*/

color: white;

padding: 5px;

border-radius: 5px;

font-size: 12px;

}

QPushButton:hover {

background-color: #8ecae6; /\* 悬停时变深 \*/

}

)");

economyClassButton->setFixedWidth(80);

economyClassLayout->addWidget(economyClassLabel);

economyClassLayout->addWidget(economyClassButton);

detailLayout->addLayout(economyClassLayout);

detailFrame->setLayout(detailLayout);

ticketLayout->addWidget(headerLabel); // 将 headerLabel 添加到 ticketFrame 布局中

ticketLayout->addWidget(detailFrame); // 将 detailFrame 添加到 ticketFrame 布局中

// 信号槽连接：点击时间与航班号标签

connect(headerLabel, &ClickableLabel::clicked, [ticketFrame, detailFrame]() {

bool isVisible = detailFrame->isVisible();

detailFrame->setVisible(!isVisible);

int newHeight = isVisible ? 60 : 140; // 展开后高度调整

ticketFrame->setFixedHeight(newHeight);

});

// 在while循环内部直接处理改签逻辑

connect(firstClassButton, &QPushButton::clicked, [this, orderNumber, flightNumber, newflightNumber, ExpensiveOrNot, cheapOneRemaining, expensiveOneRemaining]() {

bool isFirstClass = true;

double priceDifference = 0.0; // 差价

int passengerCount = 1; // 假设每次只改签一个乘客（如果有多个乘客，需要按实际乘客数调整）

// 判断剩余票数是否足够

int remainingTickets = (ExpensiveOrNot == "1") ? expensiveOneRemaining : cheapOneRemaining;

if (remainingTickets < passengerCount) {

// 剩余票数不足，弹出提示框

QDialog \*dialog = new QDialog(changeTicketsWindow);

dialog->setWindowTitle("提示");

dialog->setFixedSize(180, 80); // 弹窗大小

QVBoxLayout \*dialogLayout = new QVBoxLayout(dialog);

// 提示信息

QLabel \*messageLabel = new QLabel("剩余票数不足，改签失败");

messageLabel->setAlignment(Qt::AlignCenter);

messageLabel->setStyleSheet("font-size: 14px;");

// 确定按钮

QPushButton \*confirmButton = new QPushButton("确认");

confirmButton->setFixedSize(60, 25);

// 按钮布局

QHBoxLayout \*buttonLayout = new QHBoxLayout();

buttonLayout->addStretch();

buttonLayout->addWidget(confirmButton);

buttonLayout->addStretch();

// 添加到弹窗布局

dialogLayout->addWidget(messageLabel);

dialogLayout->addLayout(buttonLayout);

// 确认按钮功能

connect(confirmButton, &QPushButton::clicked, [this, dialog]() {

dialog->close(); // 关闭弹窗

changeTicketsWindow->close();

this->show(); // 显示主窗口

});

dialog->exec(); // 显示弹窗

} else {

// 获取当前航班的票价

QSqlQuery currentFlightQuery;

currentFlightQuery.prepare("SELECT flightExpensiveMoney, flightCheapMoney FROM flights WHERE flightNumber = ?");

currentFlightQuery.addBindValue(flightNumber); // 当前航班号

if (!currentFlightQuery.exec()) {

qDebug() << "Error fetching current flight price: " << currentFlightQuery.lastError().text();

return;

}

double currentFlightPrice = 0;

if (currentFlightQuery.next()) {

// 根据舱位类型选择对应的票价

currentFlightPrice = (ExpensiveOrNot == "1") ? currentFlightQuery.value(0).toDouble() : currentFlightQuery.value(1).toDouble();

}

// 获取改签航班的票价

QSqlQuery newFlightQuery;

newFlightQuery.prepare("SELECT flightExpensiveMoney, flightCheapMoney FROM flights WHERE flightNumber = ?");

newFlightQuery.addBindValue(newflightNumber); // 改签后的航班号

if (!newFlightQuery.exec()) {

qDebug() << "Error fetching new flight price: " << newFlightQuery.lastError().text();

return;

}

double newFlightPrice = 0;

if (newFlightQuery.next()) {

// 根据目标航班的舱位类型选择对应的票价

newFlightPrice = newFlightQuery.value(0).toDouble();

}

// 计算差价

priceDifference = newFlightPrice - currentFlightPrice;

// 更新新航班剩余票数

QSqlQuery newqueryUpdate;

newqueryUpdate.prepare("UPDATE flights SET expensiveOneRemaining = ?, cheapOneRemaining = ? WHERE flightNumber = ?");

newqueryUpdate.addBindValue(expensiveOneRemaining - 1); // 改签后目标航班头等舱剩余票数减少

newqueryUpdate.addBindValue(cheapOneRemaining); // 便宜舱位票数不变

newqueryUpdate.addBindValue(newflightNumber); // 更新的航班号

if (!newqueryUpdate.exec()) {

qDebug() << "Error updating flight ticket count: " << newqueryUpdate.lastError().text();

return;

}

// 更新原航班剩余票数

QSqlQuery queryUpdate;

if (ExpensiveOrNot == "1") {

queryUpdate.prepare("UPDATE flights SET expensiveOneRemaining = expensiveOneRemaining+1 WHERE flightNumber = ?");

} else {

queryUpdate.prepare("UPDATE flights SET cheapOneRemaining = cheapOneRemaining+1 WHERE flightNumber = ?");

}

queryUpdate.addBindValue(flightNumber); // 更新的航班号

if (!queryUpdate.exec()) {

qDebug() << "Error updating flight ticket count: " << queryUpdate.lastError().text();

return;

}

// 更新原订单的航班号和舱位类型

QSqlQuery queryOrderUpdate;

queryOrderUpdate.prepare("UPDATE orderlist SET flightNumber = ?, ExpensiveOrNot = ? WHERE orderNumber = ?");

queryOrderUpdate.addBindValue(newflightNumber); // 修改订单中的航班号

queryOrderUpdate.addBindValue(isFirstClass ? "1" : "0"); // 修改订单中的舱位类型

queryOrderUpdate.addBindValue(orderNumber); // 修改的订单号

if (!queryOrderUpdate.exec()) {

qDebug() << "Error updating order list: " << queryOrderUpdate.lastError().text();

return;

}

// 增加：修改订单状态为“已改签” (orderStatus = 2)

QSqlQuery queryStatusUpdate;

queryStatusUpdate.prepare("UPDATE orderlist SET orderStatus = ? WHERE orderNumber = ?");

queryStatusUpdate.addBindValue("2"); // 改签后的状态为“已改签”

queryStatusUpdate.addBindValue(orderNumber); // 修改的订单号

if (!queryStatusUpdate.exec()) {

qDebug() << "Error updating order status: " << queryStatusUpdate.lastError().text();

return;

}

// 弹出成功提示框

QDialog \*dialog = new QDialog(changeTicketsWindow);

dialog->setWindowTitle("改签成功");

dialog->setFixedSize(200, 80); // 弹窗大小

QVBoxLayout \*dialogLayout = new QVBoxLayout(dialog);

// 提示信息

QLabel \*messageLabel = new QLabel();

messageLabel->setAlignment(Qt::AlignCenter);

messageLabel->setStyleSheet("font-size: 14px;");

// 根据差价显示信息

if (priceDifference > 0) {

messageLabel->setText(QString("已补差价: <font color='green'>¥%1</font>").arg(priceDifference, 0, 'f', 2));

} else if (priceDifference < 0) {

messageLabel->setText(QString("已退差价: <font color='green'>¥%1</font>").arg(-priceDifference, 0, 'f', 2));

} else {

messageLabel->setText("没有差价需要支付");

}

// 确定按钮

QPushButton \*confirmButton = new QPushButton("确认");

confirmButton->setFixedSize(60, 25);

// 按钮布局

QHBoxLayout \*buttonLayout = new QHBoxLayout();

buttonLayout->addStretch();

buttonLayout->addWidget(confirmButton);

buttonLayout->addStretch();

// 添加到弹窗布局

dialogLayout->addWidget(messageLabel);

dialogLayout->addLayout(buttonLayout);

// 确认按钮功能

connect(confirmButton, &QPushButton::clicked, [this, dialog]() {

dialog->close(); // 关闭弹窗

changeTicketsWindow->close();

this->show(); // 显示主窗口

});

dialog->exec(); // 显示弹窗

}

});

// 在while循环内部直接处理改签逻辑

connect(economyClassButton, &QPushButton::clicked, [this, orderNumber, flightNumber, newflightNumber, ExpensiveOrNot, cheapOneRemaining, expensiveOneRemaining]() {

bool isFirstClass = false;

double priceDifference = 0.0; // 差价

int passengerCount = 1; // 假设每次只改签一个乘客（如果有多个乘客，需要按实际乘客数调整）

// 判断剩余票数是否足够

int remainingTickets = (ExpensiveOrNot == "1") ? expensiveOneRemaining : cheapOneRemaining;

if (remainingTickets < passengerCount) {

// 剩余票数不足，弹出提示框

QDialog \*dialog = new QDialog(changeTicketsWindow);

dialog->setWindowTitle("提示");

dialog->setFixedSize(180, 80); // 弹窗大小

QVBoxLayout \*dialogLayout = new QVBoxLayout(dialog);

// 提示信息

QLabel \*messageLabel = new QLabel("剩余票数不足，改签失败");

messageLabel->setAlignment(Qt::AlignCenter);

messageLabel->setStyleSheet("font-size: 14px;");

// 确定按钮

QPushButton \*confirmButton = new QPushButton("确认");

confirmButton->setFixedSize(60, 25);

// 按钮布局

QHBoxLayout \*buttonLayout = new QHBoxLayout();

buttonLayout->addStretch();

buttonLayout->addWidget(confirmButton);

buttonLayout->addStretch();

// 添加到弹窗布局

dialogLayout->addWidget(messageLabel);

dialogLayout->addLayout(buttonLayout);

// 确认按钮功能

connect(confirmButton, &QPushButton::clicked, [this, dialog]() {

dialog->close(); // 关闭弹窗

changeTicketsWindow->close();

this->show(); // 显示主窗口

});

dialog->exec(); // 显示弹窗

} else {

// 剩余票数足够，更新票数，执行改签操作

// 获取当前航班的票价

QSqlQuery currentFlightQuery;

currentFlightQuery.prepare("SELECT flightExpensiveMoney, flightCheapMoney FROM flights WHERE flightNumber = ?");

currentFlightQuery.addBindValue(flightNumber); // 当前航班号

if (!currentFlightQuery.exec()) {

qDebug() << "Error fetching current flight price: " << currentFlightQuery.lastError().text();

return;

}

double currentFlightPrice = 0;

if (currentFlightQuery.next()) {

// 根据舱位类型选择对应的票价

currentFlightPrice = (ExpensiveOrNot == "1") ? currentFlightQuery.value(0).toDouble() : currentFlightQuery.value(1).toDouble();

}

// 获取改签航班的票价

QSqlQuery newFlightQuery;

newFlightQuery.prepare("SELECT flightExpensiveMoney, flightCheapMoney FROM flights WHERE flightNumber = ?");

newFlightQuery.addBindValue(newflightNumber); // 改签后的航班号

if (!newFlightQuery.exec()) {

qDebug() << "Error fetching new flight price: " << newFlightQuery.lastError().text();

return;

}

double newFlightPrice = 0;

if (newFlightQuery.next()) {

// 根据目标航班的舱位类型选择对应的票价

newFlightPrice = newFlightQuery.value(1).toDouble();

}

// 计算差价

priceDifference = newFlightPrice - currentFlightPrice;

// 更新新航班剩余票数

QSqlQuery newqueryUpdate;

newqueryUpdate.prepare("UPDATE flights SET expensiveOneRemaining = ?, cheapOneRemaining = ? WHERE flightNumber = ?");

newqueryUpdate.addBindValue(expensiveOneRemaining); // 头等舱票数不变

newqueryUpdate.addBindValue(cheapOneRemaining - 1); // 改签后目标航班经济舱剩余票数减少

newqueryUpdate.addBindValue(newflightNumber); // 更新的航班号

if (!newqueryUpdate.exec()) {

qDebug() << "Error updating flight ticket count: " << newqueryUpdate.lastError().text();

return;

}

// 更新原航班剩余票数

QSqlQuery queryUpdate;

if (ExpensiveOrNot == "1") {

queryUpdate.prepare("UPDATE flights SET expensiveOneRemaining = expensiveOneRemaining+1 WHERE flightNumber = ?");

} else {

queryUpdate.prepare("UPDATE flights SET cheapOneRemaining = cheapOneRemaining+1 WHERE flightNumber = ?");

}

queryUpdate.addBindValue(flightNumber); // 更新的航班号

if (!queryUpdate.exec()) {

qDebug() << "Error updating flight ticket count: " << queryUpdate.lastError().text();

return;

}

// 更新原订单的航班号和舱位类型

QSqlQuery queryOrderUpdate;

queryOrderUpdate.prepare("UPDATE orderlist SET flightNumber = ?, ExpensiveOrNot = ? WHERE orderNumber = ?");

queryOrderUpdate.addBindValue(newflightNumber); // 修改订单中的航班号

queryOrderUpdate.addBindValue(isFirstClass ? "1" : "0"); // 修改订单中的舱位类型

queryOrderUpdate.addBindValue(orderNumber); // 修改的订单号

if (!queryOrderUpdate.exec()) {

qDebug() << "Error updating order list: " << queryOrderUpdate.lastError().text();

return;

}

// 增加：修改订单状态为“已改签” (orderStatus = 2)

QSqlQuery queryStatusUpdate;

queryStatusUpdate.prepare("UPDATE orderlist SET orderStatus = ? WHERE orderNumber = ?");

queryStatusUpdate.addBindValue("2"); // 改签后的状态为“已改签”

queryStatusUpdate.addBindValue(orderNumber); // 修改的订单号

if (!queryStatusUpdate.exec()) {

qDebug() << "Error updating order status: " << queryStatusUpdate.lastError().text();

return;

}

// 弹出成功提示框

QDialog \*dialog = new QDialog(changeTicketsWindow);

dialog->setWindowTitle("改签成功");

dialog->setFixedSize(200, 80); // 弹窗大小

QVBoxLayout \*dialogLayout = new QVBoxLayout(dialog);

// 提示信息

QLabel \*messageLabel = new QLabel();

messageLabel->setAlignment(Qt::AlignCenter);

messageLabel->setStyleSheet("font-size: 14px;");

// 根据差价显示信息

if (priceDifference > 0) {

messageLabel->setText(QString("已补差价: <font color='green'>¥%1</font>").arg(priceDifference, 0, 'f', 2));

} else if (priceDifference < 0) {

messageLabel->setText(QString("已退差价: <font color='green'>¥%1</font>").arg(-priceDifference, 0, 'f', 2));

} else {

messageLabel->setText("没有差价需要支付");

}

// 确定按钮

QPushButton \*confirmButton = new QPushButton("确认");

confirmButton->setFixedSize(60, 25);

// 按钮布局

QHBoxLayout \*buttonLayout = new QHBoxLayout();

buttonLayout->addStretch();

buttonLayout->addWidget(confirmButton);

buttonLayout->addStretch();

// 添加到弹窗布局

dialogLayout->addWidget(messageLabel);

dialogLayout->addLayout(buttonLayout);

// 确认按钮功能

connect(confirmButton, &QPushButton::clicked, [this, dialog]() {

dialog->close(); // 关闭弹窗

changeTicketsWindow->close();

this->show(); // 显示主窗口

});

dialog->exec(); // 显示弹窗

}

});

// 将票务框架添加到布局中

ticketListLayout->addWidget(ticketFrame);

}

// 将滚动内容绑定到滚动区域

scrollWidget->setLayout(ticketListLayout);

scrollArea->setWidget(scrollWidget);

// 添加滚动区域到主布局

mainLayout->addWidget(scrollArea);

// 返回按钮布局

QHBoxLayout \*buttonLayout = new QHBoxLayout();

QPushButton \*returnButton = new QPushButton("返回");

returnButton->setStyleSheet(R"(

QPushButton {

background-color: #219ebc; /\* 按钮默认背景颜色：浅蓝色 \*/

color: white; /\* 按钮字体颜色 \*/

padding: 2px 15px; /\* 按钮内边距 \*/

border-radius: 5px; /\* 圆角边框 \*/

font-size: 14px; /\* 字体大小 \*/

}

QPushButton:hover {

background-color: #8ecae6; /\* 悬停时的颜色：浅蓝色更深一些 \*/

}

QPushButton:pressed {

background-color: #006f79; /\* 按下时的颜色：深蓝色 \*/

}

)");

buttonLayout->addStretch();

buttonLayout->addWidget(returnButton);

buttonLayout->addStretch();

mainLayout->addLayout(buttonLayout);

// 返回按钮功能

connect(returnButton, &QPushButton::clicked, [this]() {

changeTicketsWindow->close();

this->show(); // 显示主窗口

});

// 设置主布局并显示窗口

changeTicketsWindow->setLayout(mainLayout);

changeTicketsWindow->show();

}