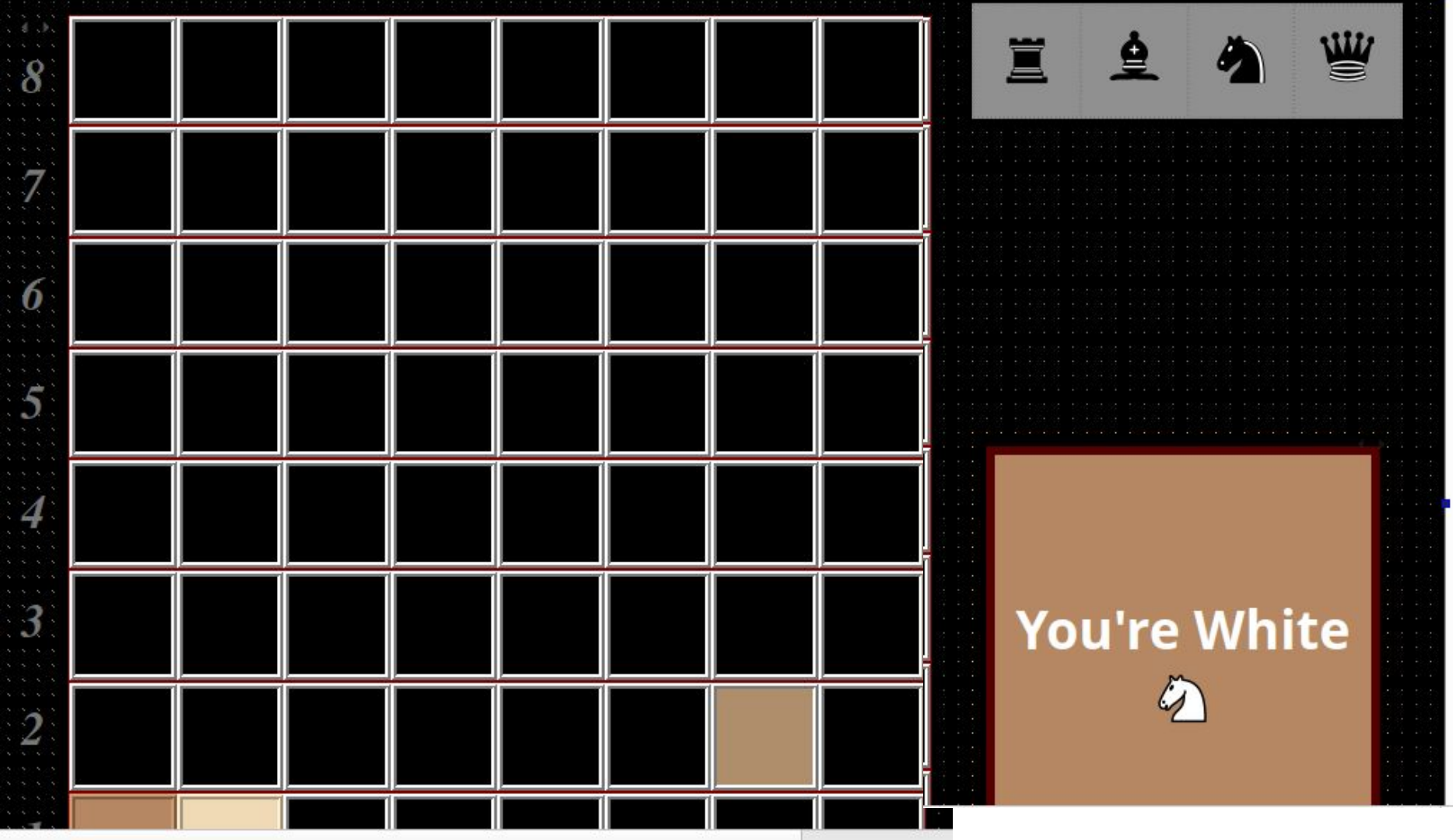

Schach


8
7
6
5
4
3
2
You're White

Button Array in the constructor

```
button[0][0] = std::pair<QPushButton*, QLabel*>(ui->fb_0,ui->P_0);
 button[0][1] = std::pair<QPushButton*, QLabel*>(ui->fb_1,ui->P_1);
 button[0][2] = std::pair<QPushButton*, QLabel*>(ui->fb_2,ui->P_2);
 button[0][3] = std::pair<QPushButton*, QLabel*>(ui->fb_3,ui->P_3);
[...]
 button[7][5] = std::pair<QPushButton*, QLabel*>(ui->fb_61,ui->P_61);
 button[7][6] = std::pair<QPushButton*, QLabel*>(ui->fb_62,ui->P_62);
button[7][7] = std::pair<QPushButton*, QLabel*>(ui->fb_63,ui->P_63);
```

```
void Board::startBoard(int type){
    if(type == 1) {
        ui->Display->setCurrentIndex(1);
        ui->Rows->setCurrentIndex(1);
        for (int i = 0; i < 4; ++i) {
            for(int j = 0; j < 8; j++){
                //QString tmp = button[i][j].first->parentWidget()->styleSheet();
                //QPixmap img = *(button[i][j].second->pixmap());
                std::pair<QPushButton*, QLabel*> cmp = button[i][j];
                //button[i][j].second->setPixmap(*(button[7-i][j].second->pixmap()));
                //button[7-i][j].second->setPixmap(img);
                //button[i][j].first->parentWidget()->setStyleSheet(button[7-i][j].first->parentWidget()->styleSheet());
                //button[7-i][j].first->parentWidget()->setStyleSheet(tmp);
                button[i][j]=button[7-i][j];
                button[7-i][j] = cmp;
            }}}
      else{
        ui->Display->setCurrentIndex(0);
        ui->Rows->setCurrentIndex(0);
        }

    //Promotion prompts for gameplay. Hidden at the start.
    ui->promPrompt->hide();
    ui->promPrompt_2->hide();

    //Rematch Button. Hidden at the start.
    ui->reMatch->hide();

    //Colouring the fields
    for (int i = 0; i < 8; ++i) {
        for(int j = 0; j < 8; j++){
            button[i][j].first->setStyleSheet(invisButt);
            if((i+j)%2==0){
                button[i][j].first->parentWidget()->setStyleSheet(hellsetting);
            }else{
                button[i][j].first->parentWidget()->setStyleSheet(darksetting);
            }
        } }
```

```
120     for (int i = 0; i < 8; ++i) {
121         for(int j = 0; j < 8; j++){
122             button[i][j].first->setStyleSheet(invisButt);
123             if((i+j)%2==0){
124                 button[i][j].first->parentWidget()->setStyleSheet(hellsetting);
125             }else{
126                 button[i][j].first->parentWidget()->setStyleSheet(darksetting);
127             }
128         } }
129
130     //Set the Pieces
131     button[0][0].second->setPixmap(WRook); //Rooks
132     button[0][7].second->setPixmap(WRook);
133     button[7][0].second->setPixmap(BRook);
134     button[7][7].second->setPixmap(BRook);
135     button[0][1].second->setPixmap(WKnight);//Knights
136     button[0][6].second->setPixmap(WKnight);
137     button[7][1].second->setPixmap(BKnight);
138     button[7][6].second->setPixmap(BKnight);
139     button[0][2].second->setPixmap(WBishop); //Bishops
140     button[0][5].second->setPixmap(WBishop);
141     button[7][2].second->setPixmap(BBishop);
142     button[7][5].second->setPixmap(BBishop);
143     button[0][3].second->setPixmap(WQueen); //Queens
144     button[7][3].second->setPixmap(BQueen);
145     button[0][4].second->setPixmap(WKing); //Kings
146     button[7][4].second->setPixmap(BKing);
147     for (int j = 0; j < 8; j++) { //Pawns
148         button[1][j].second->setPixmap(WPawn);
149         button[6][j].second->setPixmap(BPawn);
150     }
151
152     //Connect the buttons
153     for (int i = 0; i < 8; ++i) {
154         for(int j = 0; j < 8; j++){
155             connect(button[i][j].first,
156                     &QPushButton::clicked,
157                     this,
158                     [=](){
159                 emit Buttonpos(i,j);
160                 qDebug()<<"Button ["<<i<<"]["<<j<<"] is working\n";});}
```

```
204
205 /**Slot that moves pieces by changing the QLabels. Also removes highlighting and empties framers afterwards.
206 Also acts as a Capture method since what it does is:
207                                     1) change the Label of toX, toY position to the Label of the First.
208                                     2) Change the Label of the fromX, fromY position to an empty png.**/
209 void Board::moveLabel(int fromX, int fromY, int toX, int toY, int _enpassant){
210     button[toX][toY].second->setPixmap(*(button[fromX][fromY].second->pixmap()));
211     button[fromX][fromY].second->setPixmap(Nul0);
212     if(_enpassant == 1){
213         if(toX-fromX == 1)
214             button[toX-1][toY].second->setPixmap(Nul0);
215         else
216             button[toX+1][toY].second->setPixmap(Nul0);
217     }
218     if(m_highlighted){
219         std::for_each(
220                     framers.begin(),
221                     framers.end(),
222                     [&](std::pair<int, int> a){
223             lightUp(a.first, a.second);
224             framers.clear();}
225         );
226         m_highlighted = false;}
227 }
228
229 /*Slot that removes highlighting using lightUp in case there's no movement.
230     And also empties the framers vector.*/
231 void Board::NoMove(){
232     if(m_highlighted){
233         std::for_each(
234                     framers.begin(),
235                     framers.end(),
236                     [&](std::pair<int, int> a){
237             lightUp(a.first, a.second);
238             framers.clear();}
239         );
240         m_highlighted = false;}
241 }
242
243
244 /*Prompt to show pieces that can be promoted to
```

```
void Board::Promote(int x, int y, int type){
    qDebug()<<"Doing STUFF!!";
    qDebug()<<"The points are"<<x<<","<<y;
    ui->ChessBoard->setEnabled(true);
    if (x==0)
    {ui->promPrompt_2->hide();}else{ui->promPrompt->hide();}
    switch (type) {
    case 4:
        qDebug()<<"Case 4 is working!!!";
        if(x==0){
            button[x][y].second->setPixmap(*(ui->BBae->pixmap()));
        }else{
            button[x][y].second->setPixmap(*(ui->WBae->pixmap()));
        }
        break;
    case 3:
        qDebug()<<"Case 3 is working!!!";
        if(x==0){
            button[x][y].second->setPixmap(*(ui->BBurg->pixmap()));
        }else{
            button[x][y].second->setPixmap(*(ui->WBurg->pixmap()));
        }
        break;
    case 2:
        qDebug()<<"Case 2 is working!!!";
        if(x==0){
            button[x][y].second->setPixmap(*(ui->BKrieg->pixmap()));
        }else{
            button[x][y].second->setPixmap(*(ui->WKrieg->pixmap()));
        }
        break;
    case 1:
        qDebug()<<"Case 1 is working!!!";
        if (x==0){
            button[x][y].second->setPixmap(*(ui->BPope->pixmap()));
        }else{
            button[x][y].second->setPixmap(*(ui->WPope->pixmap()));
        }
        break;
    default:
```


Address
Port
Server
Client
Initialize
Close
Connect
Disconnect
Vertical Spacer 'verticalSpacer', 20 x 40


Form
Server
Client

```cpp
#include "interface.h"
#include "ui_interface.h"
#include <QDebug>

Interface::Interface(QWidget *parent) :
    QWidget(parent),
    ui(new Ui::Interface)
{
    ui->setupUi(this);

    //for
    startBut = ui->Start;
    endBut = ui->Stop;
    conBut = ui->connButt;
    disCon = ui->DiscButt;
    ui->connButt->hide();
    ui->Start->hide();
    ui->Stop->hide();
    //ui->Stop->setEnabled(false);
    ui->DiscButt->hide();
    ui->Addr_Port->hide();
```

Logic

```
818 void Gameboard::knightMoves(int x, int y){
819
820     int knightX[8] = { 2, 1, -1, -2, -2, -1, 1, 2}; //for knight
821     int knightY[8] = { 1, 2, 2, 1, -1, -2, -2, -1}; //for knight
822
823     int tmpX = x;
824     int tmpY = y;
825     auto myColor = board[tmpX][tmpY]->getColor();
826
827
828     for(int i = 0; i < 8; i++){
829         x = tmpX + knightX[i];
830         y = tmpY + knightY[i];
831
832         if(x >= 0 && y >= 0 && x < 8 && y < 8)
833         {
834             if(myColor != board[x][y]->getColor()){
```

```
836                 auto a = board[x][y];
837                 board[x][y] = board[tmpX][tmpY];
838                 board[tmpX][tmpY] = std::make_shared<NullPiece>(NullPiece());
839                 getKingPosition();
840                 getVirtualThreats();
841                 if(!(virtualGrid[_kingPosition.first][_kingPosition.second])){
842 
843                     _legalMoves.push_back(std::pair<int,int>(x,y));
844                     //resetVirtualGrid();
845                 }
846 
847                 //reverse mock move
848                 board[tmpX][tmpY] = board[x][y];
849                 board[x][y] = a;
850             }
851 
852         }
853     }
854 }
```

```
254 void Gameboard::hasGameEnded(){
255     size_t legalMoveCount = 0;
256     int knightCount = 0;
257     int bishopCount = 0;
258     int pieceCount = 0;
259 
260     for(int i = 0; i < 8; i++){
261         for(int j = 0; j < 8; j++){
262             if(board[i][j]->getType() != empty){
263                 pieceCount++;
264                 if(board[i][j]->getType() == knight){
265                     knightCount++;
266                 }
267                 else if(board[i][j]->getType() == bishop){
268                     bishopCount++;
269                 }
270             }
271         }
272     }
273     if(pieceCount == 3){
274         if(bishopCount == 1 || knightCount == 1){
275             qDebug() << "insufficient material";
276             emit endGame(3,0);   //insufficient material
277             _gameEnded = true;
278             return;
279         }
280     }
```

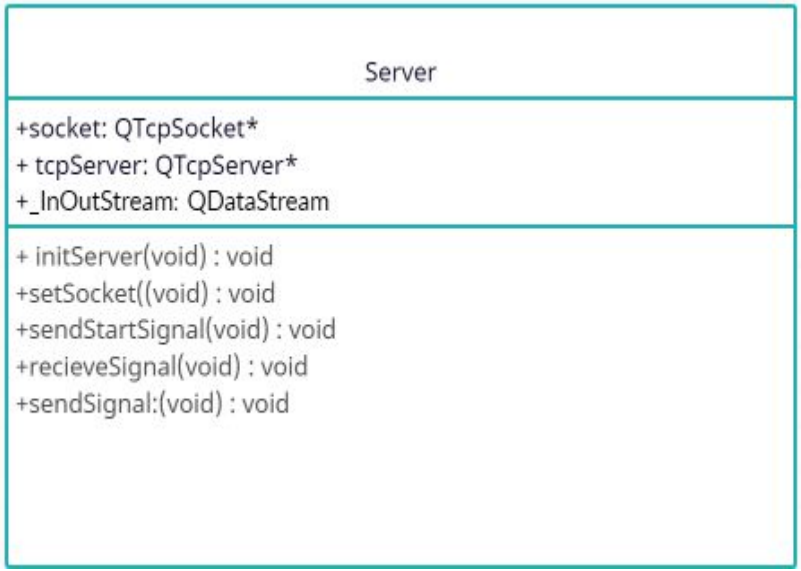
```
281     else if(pieceCount == 2){
282         qDebug() << "insufficient material";
283         emit endGame(3,0);      //insufficient material
284         _gameEnded = true;
285         return;
286     }
287     else{
288         for(int i = 0; i < 8; i++){
289             for(int j = 0; j < 8; j++){
290                 if(board[i][j]->getColor() == white && _turn  == w){
291                     legalMoveCount += getLegalMoves(i,j).size();
292                     if(legalMoveCount > 0)
293                         return;
294                 }
295                 else if(board[i][j]->getColor() == black && _turn  == b){
296                     legalMoveCount += getLegalMoves(i,j).size();
297                     if(legalMoveCount > 0)
298                         return;
299                 }
300 
301             }
302         }
303         qDebug() << legalMoveCount;
304 
305         if(_kingChecked){
306             qDebug() << "Checkmate! " << _turn << " lost";
307             emit endGame(1, (int)_turn); //checkmate
308             _gameEnded = true;
309             return;
310 
311         }
```

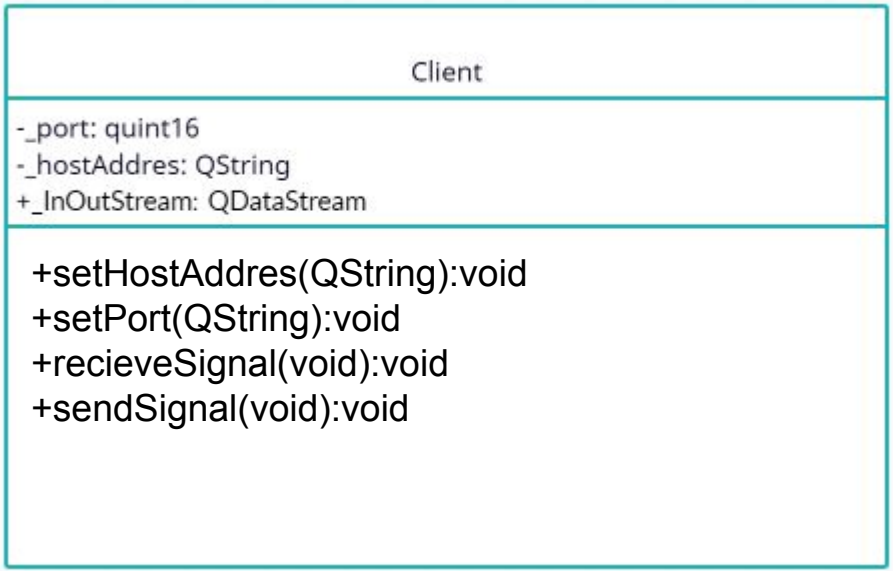
```
        }
        else{
            qDebug() << "haha stalemate" ;
            emit endGame(2, 0); //stalemate, turn not important while emitting
            _gameEnded = true;
            return;
                    }
    }
```

```
115
116 void Gameboard::enPassant(int fromX, int fromY, int toX, int toY){
117
118     if(board[toX][toY]->getType() == pawn && ((fromX == toX - 2) || (fromX == toX + 2))){
119
120         if(board[toX][toY+1]->getType() == pawn)
121             board[toX][toY+1]->_enpassantLeft = true;
122         if(board[toX][toY-1]->getType() == pawn)
123             board[toX][toY-1]->_enpassantRight = true;
124     }
125 }
126
127 void Gameboard::revokeEnpassant(){
128     for(int i = 0; i <8; i++){
129         for(int j = 0; j < 8; j++){
130             board[i][j]->_enpassantLeft = false;
131             board[i][j]->_enpassantRight = false;
132         }
133     }
134 }
135
```

```
696             if(board[x+1][y+1]->getColor() == black || board[x][y]->_enpassantRight){
697
698                 auto a = board[x+1][y+1];
699                 board[x+1][y+1] = board[x][y];
700                 board[x][y] = std::make_shared<NullPiece>(NullPiece());
701                 getKingPosition();
702                 getVirtualThreats();
703                 if(!(virtualGrid[_kingPosition.first][_kingPosition.second])){
704
705                     _legalMoves.push_back(std::pair<int,int>(x+1,y+1));
706                     //resetVirtualGrid();
707                 }
708
709                 //reverse mock move
710                 board[x][y] = board[x+1][y+1];
711                 board[x+1][y+1] = a;
712             }
713         }
```

Network UML


Server
+socket: QTcpSocket*
+ tcpServer: QTcpServer*
+_InOutStream: QDataStream
+ initServer(void) : void
+setSocket((void) : void
+sendStartSignal(void) : void
+recieveSignal(void) : void
+sendSignal:(void) : void


Client
-_port: quint16
-_hostAddres: QString
+_InOutStream: QDataStream
+setHostAddres(QString):void
+setPort(QString):void
+recieveSignal(void):void
+sendSignal(void):void

Netzwerk

```
10  void Server::initServer(){
11
12
13      tcpServer = new QTcpServer(this);
14
15      quint16 port = 1234;
16
17
18      tcpServer->setMaxPendingConnections(1);
19      connect(tcpServer,SIGNAL(newConnection()),this,SLOT(setSocket()));
20
21      tcpServer->listen(QHostAddress::Any,port);
22      if(tcpServer->isListening())
23       qDebug()<<"server is listening on port: 1234";
24      else
25          qDebug()<<"server isn't listening";
26
27
28  }
```

Netzwerk

```
139 void Server::setSocket(){
140     socket = tcpServer->nextPendingConnection();
141     _InOutStream.setDevice(socket);
142     connect(socket,SIGNAL(readyRead()),this,SLOT(recieveSignal()));
143     connect(socket,SIGNAL(error(QAbstractSocket::SocketError)),this,SLOT(socketError()));
144     connect(socket,SIGNAL(destroyed()),this,SLOT(closeConnection()));
145     connect(socket,SIGNAL(disconnected()),this,SLOT(closeConnection()));
146     sendStartSignal();
147     tcpServer->close();
148     }
```

Netzwerk

```
120 void Client::sendSignal(quint8 cmd,quint8 startRow ,
121                        quint8 startColumn ,quint8 targetRow ,quint8 targetColumn,quint8 promotionType,quint8 concequense ,quint8 status){
122     quint8 groupNum  = 1;
123     quint8 length;
124     switch (cmd) {
125
126     case 2:
127     {
128         length = 1;
129
130         _InOutStream << cmd << length << groupNum;
131         qDebug()<<"client sent 2 1 1";
132         break;
133     }
134     case 3:
135     {
136         length = 5;
137         quint8 extraInfo = ((promotionType<<4)&0xf0) | (concequense & 0x0f);
138
139         _InOutStream << cmd << length << startColumn << startRow << targetColumn << targetRow << extraInfo;
140         qDebug()<<"server sent: "<<cmd << length << startColumn << startRow << targetColumn << targetRow << extraInfo ;
141         break;
142     }
143     case 4:
144     {
145         length = 1;
146         _InOutStream << cmd << length << status ;
147         break;
148     }
149     default:
150     {
151         break;
152     }
153     }
154  }
155
```