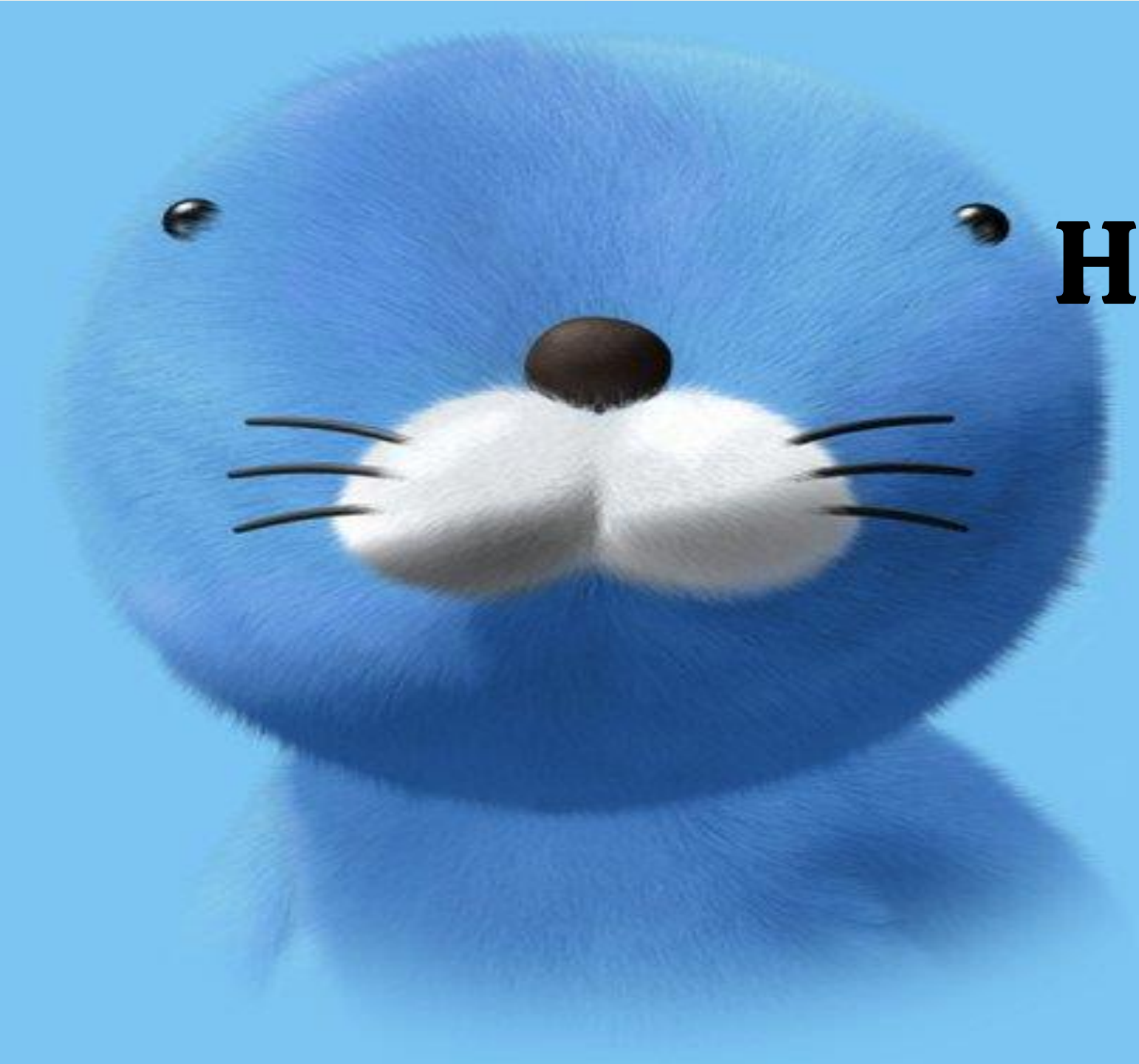
ENTERPARK TICKET

ENTERPARK TICKET

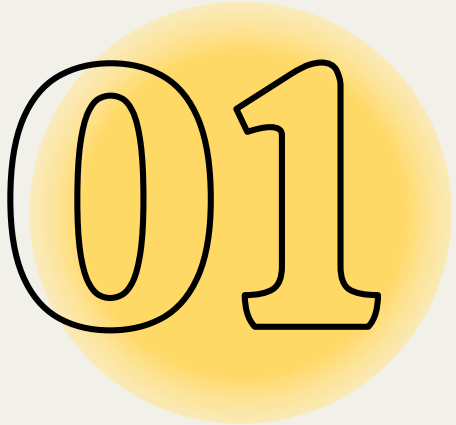
성공적인 티켓팅을 위하여



Hello! We are...

Coding Breakdown Team

ENTERPARK of Contents.



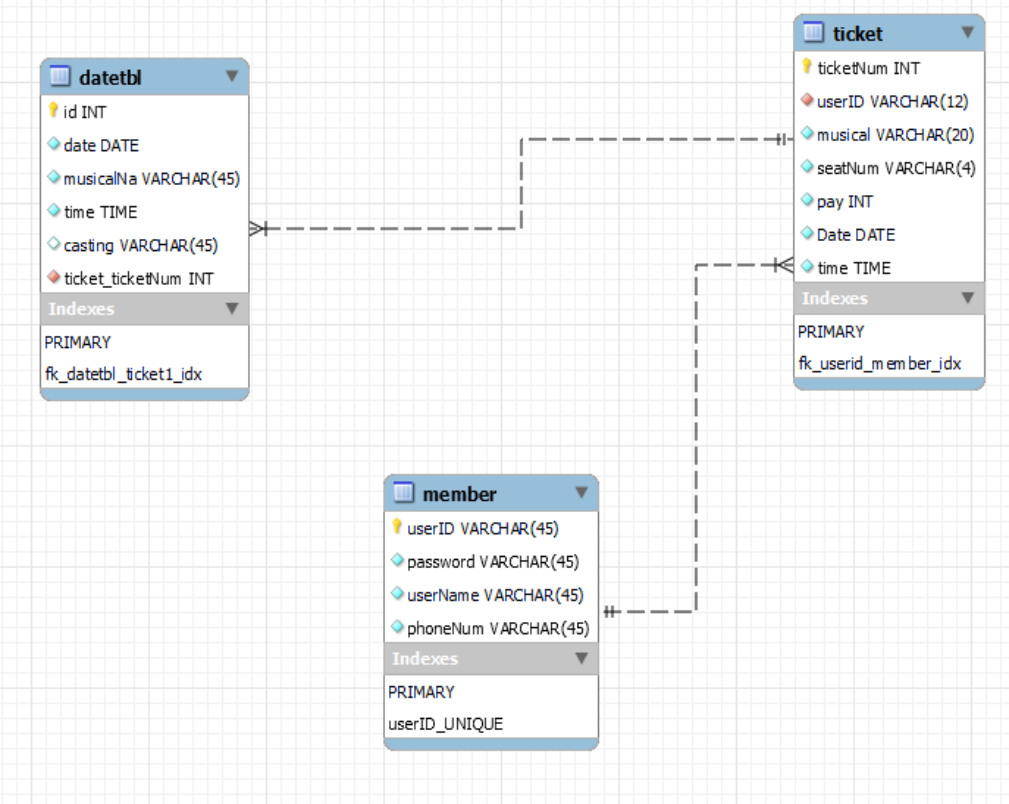
We will talk about this first.

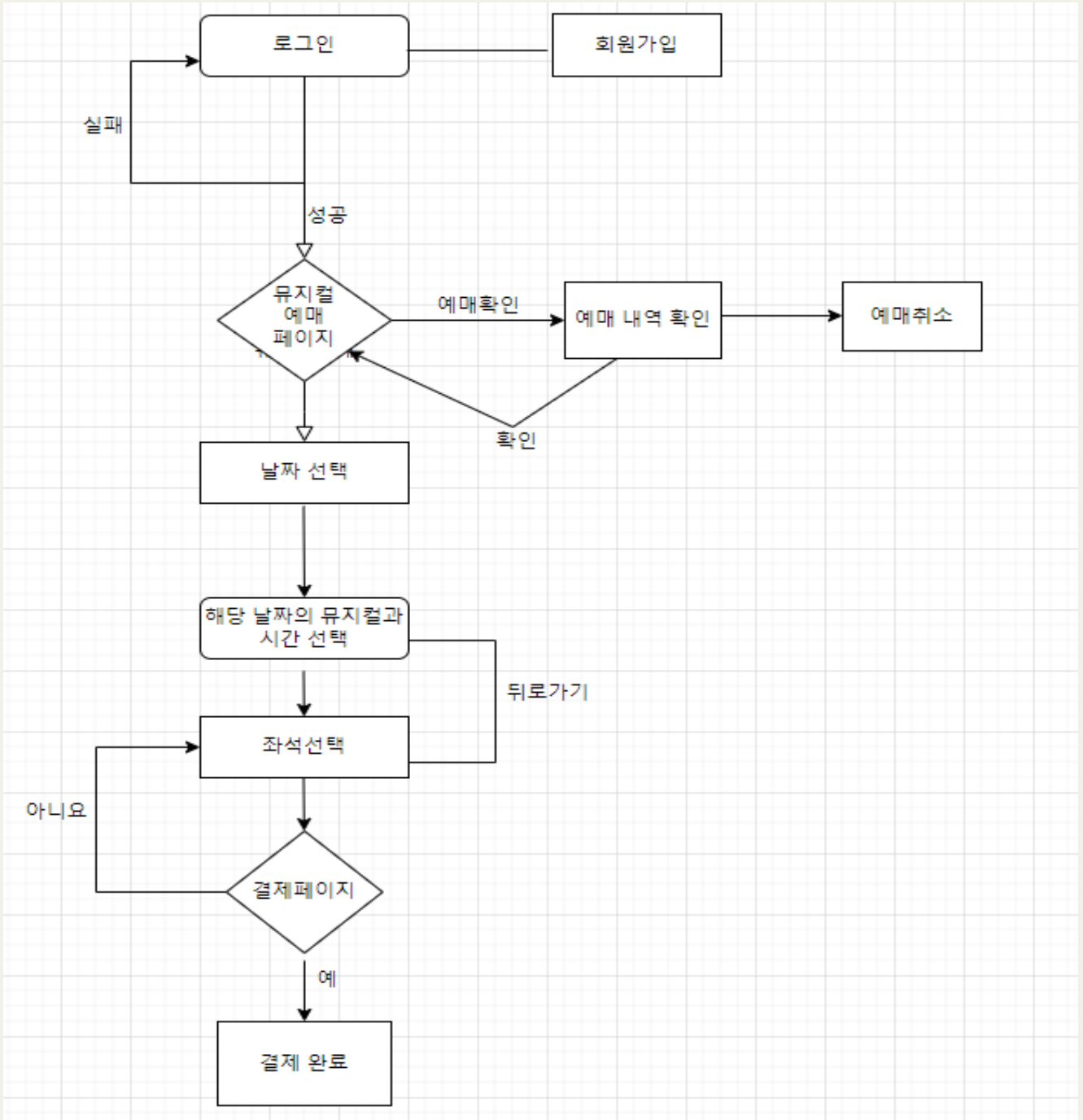
개요

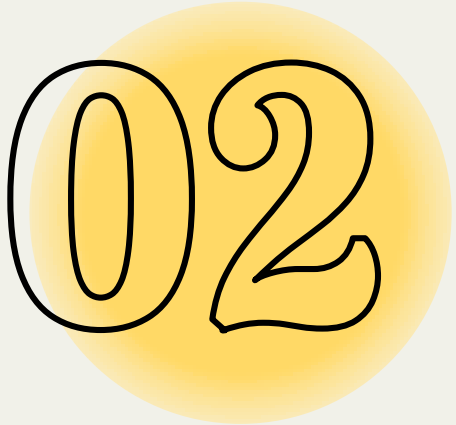
개요

- Java와 javafx를 통해 뮤지컬 티켓 예매 프로그램 구현
- MySQL이용한 데이터베이스 연동
- 서버, 클라이언트 구현 및 네트워크 이해
- 프로젝트 진행을 통해 팀워크 및 협업 능력 향상

화면 구성







We will talk about this second.

팀 소개 및 역할

This is our team.

박 종현

서버, 클라이언트 구축
DB 연동
결제 /예매 확인

박 진성

서버, 클라이언트 구축
DB 연동
좌석선택

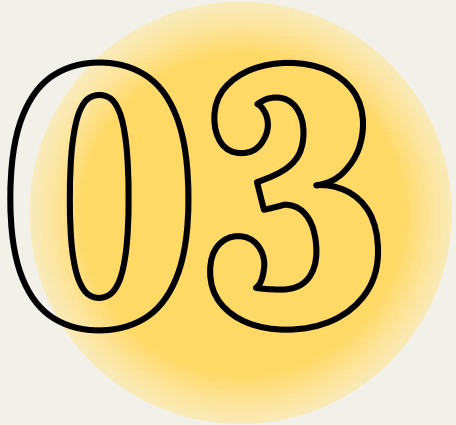
엄 수연

서버, 클라이언트 구축
DB 연동
예약화면/ 예매 내역

정 애란

서버, 클라이언트 구축
DB 연동
로그인/ 회원가입

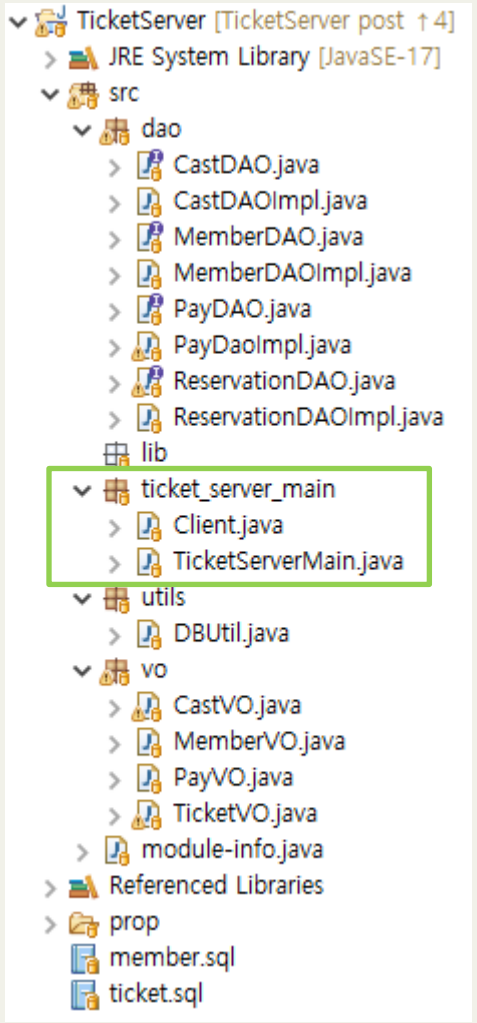
This is a timeline.



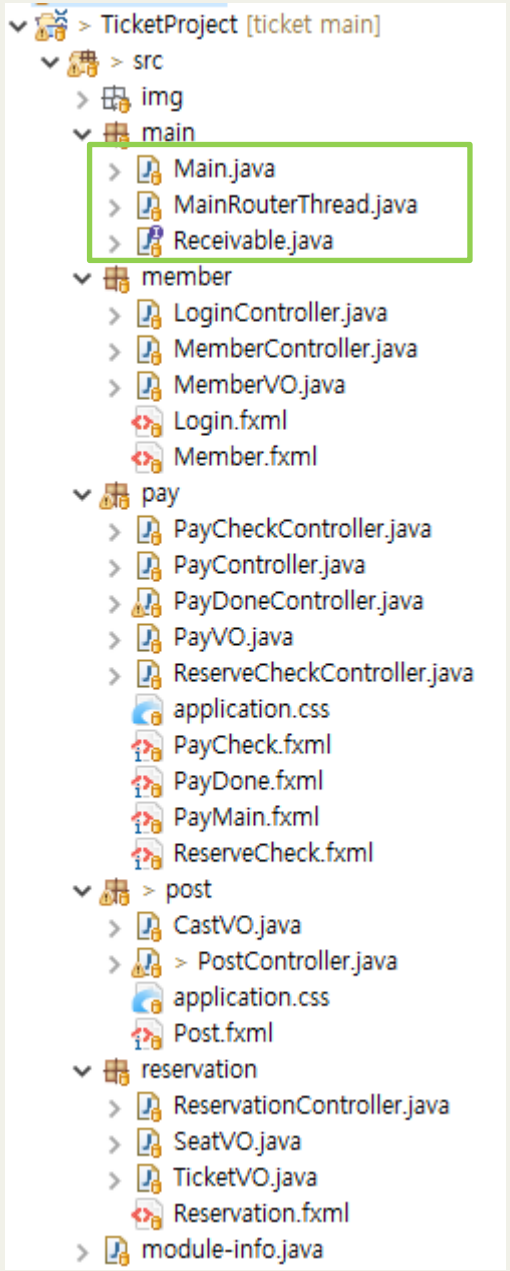
We will talk about this third.

수행 절차 및 방법

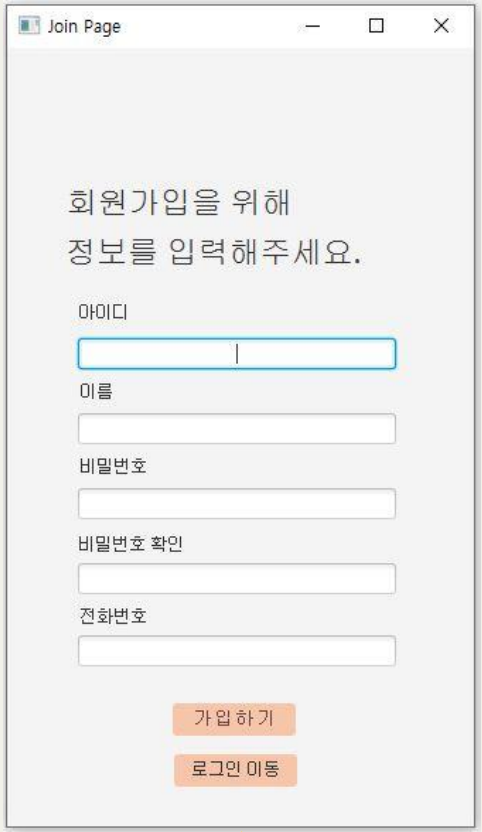
Server



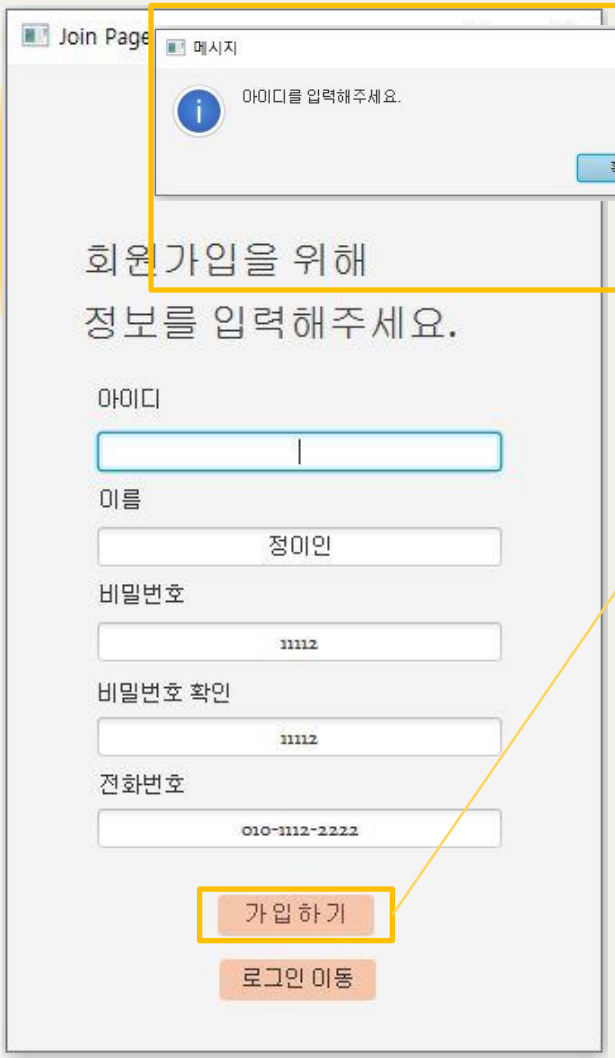
Client




```
btnLogin.setOnAction(e -> login());

btnMember.setOnAction(e->{
    try {
        // 회원 페이지 이동
        Stage stage = new Stage();
        Parent root = FXMLLoader.load(getClass().getResource("/member/Member.fxml"));
        Scene scene = new Scene(root);
        stage.setScene(scene);
        stage.setTitle("Join Page");
        stage.show();

        // Login Stage close
        Stage loginStage = (Stage)btnMember.getScene().getWindow();
        loginStage.close();

    } catch (IOException e1) {
        e1.printStackTrace();
    }
});
}

public void login() {
    if (txtId.getText().isEmpty() || txtPw.getText().isEmpty()) {
        System.out.println("아이디 혹은 비밀번호를 입력하지 않았습니다!");
        txtId.clear();
        txtPw.clear();
        txtId.requestFocus();
            return;
    }
```



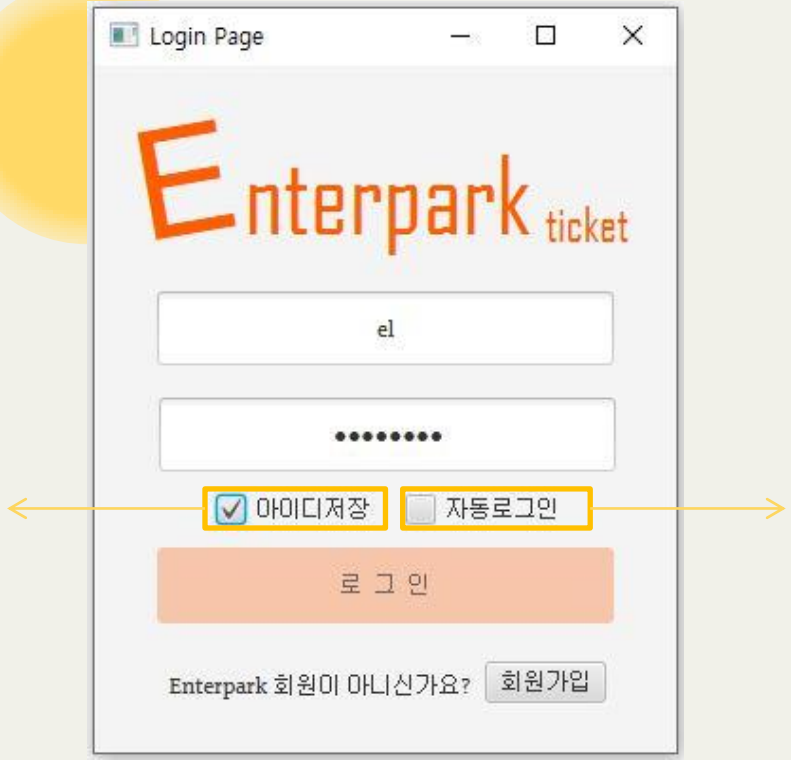
```java
if(id.length() == 0) {
    alert("아이디를 입력해주세요.");
    // 빈문자열
    txtId.requestFocus();
    return;
}
if(name.length() == 0) {
    alert("이름을 입력해주세요.");
    txtName.requestFocus();
    return;
}
if(pw.length() == 0) {
    alert("비밀번호를 입력해주세요.");
    txtPw.requestFocus();
    return;
}
if(rPw.length() == 0) {
    alert("비밀번호를 확인해주세요.");
    txtRe.requestFocus();
    return;
}
if(phone.length() == 0) {
    alert("전화번호를 입력해주세요.");
    txtPhone.requestFocus();
    return;
}

if (!rPw.equals(pw) ) {
    System.out.println("비밀번호가 일치하지 않습니다.");
    alert("비밀번호가 일치하지 않습니다.");
    return;
}

MemberVO m = new MemberVO(id,pw,name,phone);
Main.thread.sendData("0|1|"+m);
```




```java
public void alert(String text){
    Platform.runLater(()->{
        Alert alert = new Alert(AlertType.INFORMATION);
        alert.setHeaderText(null);
        alert.setContentText(text);
        alert.showAndWait();
    });
}
```

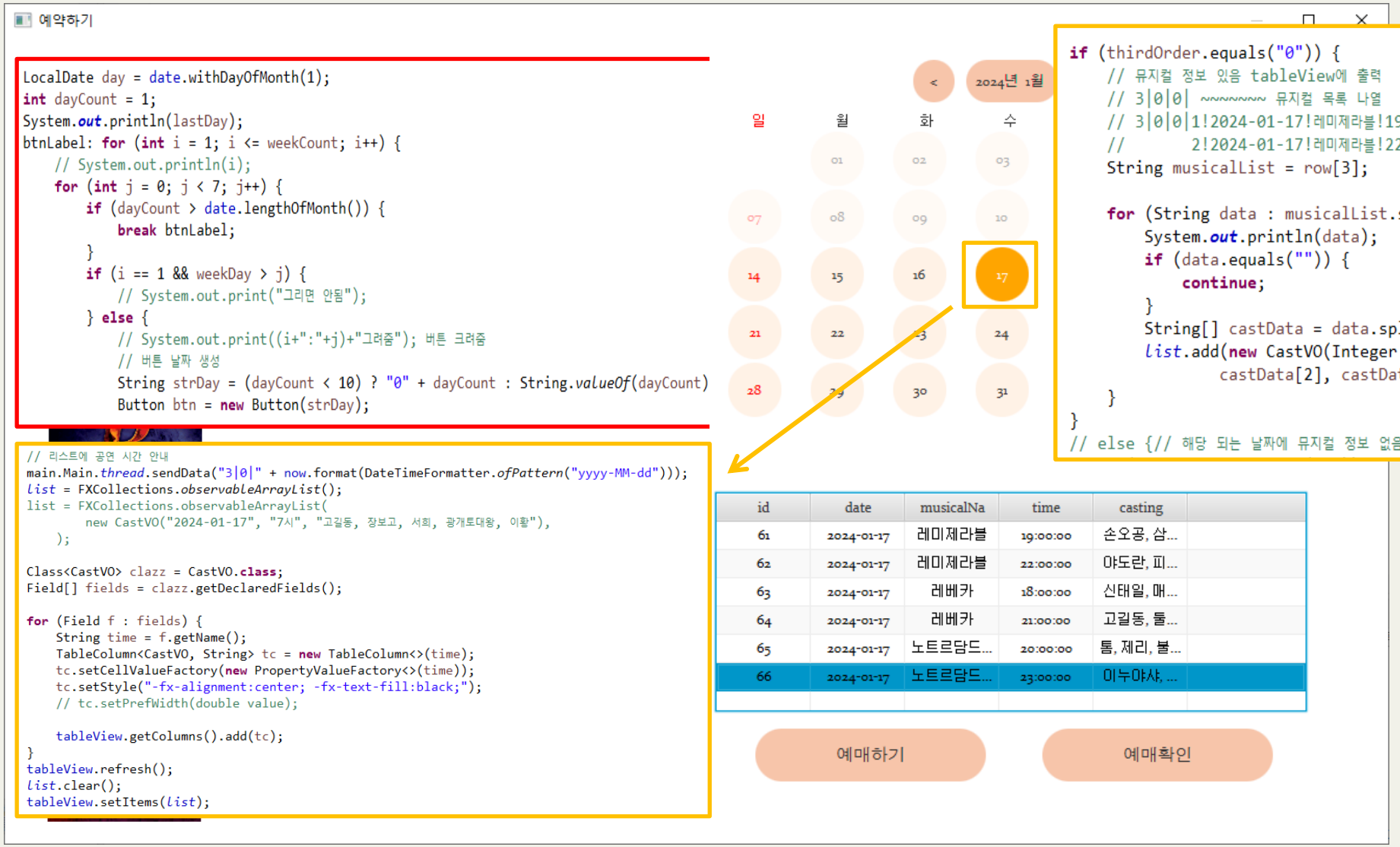
```java
@Override
public void receiveData(String message) {
    System.out.println("MemberController : " + message);
    //0|1|true, 0|1|false
    String isJoin = message.split("\\|")[2];
    if(Boolean.parseBoolean(isJoin)) {
        System.out.println("회원가입 완료");
        alert("회원가입 완료");
        // 화면 전환 -> 로그인
        Platform.runLater(()->{
            buttonGo.fire();
        });
    }else {
        // 회원가입 실패
        alert("회원가입 실패");
        Platform.runLater(()->{
            txtId.clear();
            txtPw.clear();
            txtName.clear();
            txtRe.clear();
            txtId.clear();
            txtPhone.clear();
            txtId.requestFocus();
        });
    }
}
```

```java
public void rememberUserId(String userId) {
    try {
        File dir = new File("/member");
        if(!dir.exists()) {
            dir.mkdirs();
        }
        File file = new File(dir,"rememberMe.txt");
        System.out.println(file.getAbsolutePath());
        System.out.println(file.getPath());
        FileOutputStream fos = new FileOutputStream(file);
        fos.write(userId.getBytes());
        fos.flush();
        fos.close();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
public String readUserIdWithFile() {
    File dir = new File("/member");
    File file = new File(dir,"rememberMe.txt");
    if(!dir.exists() || !file.exists()) {
        return null;
    }

    String userID = null;
    try {
        FileInputStream fis = new FileInputStream(file);
        // 연결된 파일의 읽을 수 있는 byte 개수 만큼 배열 생성
        byte[] bytes = new byte[fis.available()];
        fis.read(bytes);
        if(bytes.length > 0) {
            userID = new String(bytes);
        }
        fis.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
    return userID;
}
```




```java
Platform.runLater(()->{

    if(rememberMe.isSelected()) {
        rememberUserId(Main.loginMember.getUserID());
    }

    if(autoLogin.isSelected()){
        rememberMemberInfo();
    }
```

```java
public void rememberMemberInfo() {
    String userId = Main.loginMember.getUserID();
    String password = Main.loginMember.getPassword();

    try {
        File dir = new File("/member");
        if(!dir.exists()) {
            dir.mkdirs();
        }
        File file = new File(dir,"rememberMemberInfo.txt");
        FileOutputStream fos = new FileOutputStream(file);
        fos.write(userId.getBytes());
        fos.write(".".getBytes());
        fos.write(password.getBytes());
        fos.flush();
        fos.close();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }
}

public MemberVO rememberMeReadMemberInfo() {

    MemberVO vo = null;
    File dir = new File("/member");
    if(!dir.exists()) {
        dir.mkdirs();
    }
    File file = new File(dir,"rememberMemberInfo.txt");
    if(!dir.exists() || !file.exists()) {
        return null;
    }

  String memberInfo = null;

  try {
      FileInputStream fis = new FileInputStream(file);
      byte[] bytes = new byte[fis.available()];
      fis.read(bytes);
      if(bytes.length > 0) {
          memberInfo = new String(bytes);
          String[] results = memberInfo.split("\\.");
          if(results.length == 2) {
              vo = new MemberVO(results[0],results[1]);
          }
      }
      fis.close();
  } catch (IOException e) {
      e.printStackTrace();
  }
  return vo;
```



```java
// 왼쪽 마우스 두번 클릭
if(btn == MouseButton.PRIMARY && clickCount == 2) {
    Main.reservTicket = tableView.getSelectionModel().getSelectedItem();
    if(Main.reservTicket == null) return;

    Stage stage = new Stage(StageStyle.UTILITY);
    Parent root = null;
    FXMLLoader loader = null;

    try {
        loader = new FXMLLoader(getClass().getResource("/pay/PayCheck.fxml"));
        root = loader.load();
    } catch (IOException e1) {
        e1.printStackTrace();
        return;
```

예약하기

< 2024년 1월 >

노트르담드파리

NOTREDAMEDE
PARIS

65	2024-01-17	노트르담드...	20:00:00	톰, 제리, 불...
66	2024-01-17	노트르담드...	23:00:00	이누야샤, ...

예매하기 예매확인

예매확인

예매 내역

musical	seatNum	date	time
노트르담드...	D-9	2024-01-17	23:00:00
레베카	B-8	2024-01-11	21:00:00
레베카	B-9	2024-01-11	18:00:00
레베카	C-7	2024-01-18	18:00:00

확인

예매확인

예매 확인

o 님

예매하신 뮤지컬 이름은

노트르담드파리

좌석은

D-9

날짜는

2024-01-17 / 23:00:00

확인 예매 취소

Main.java

```
32
33      // 테이블 뷰 에서 선택된 예약 뮤지컬 시간 정보
34      public static CastVO castVO;
@Override
public void initialize(URL arg0, ResourceBundle arg1) {
    Main.thread.reservationController = this;
    selectDate.setText(Main.castVO.getDate());
    selectTime.setText(Main.castVO.getTime());
    selectMusical.setText(Main.castVO.getMusicalNa());
```

```
public void setSeats(List<String> list) {
    // 버튼 만들기
    for (int i = 1; i < 11; i++) {
        HBox hbox = new HBox();
        hbox.setPrefWidth(500);
        hbox.setSpacing(20);
        for (int j = 1; j < 11; j++) {
            int val = 64 + i;
            char value = (char) val;
            Button b = new Button(value + "-" + j);
            if(list.contains(b.getText())) {
                b.setDisable(true);
            }
            b.setMaxWidth(Double.MAX_VALUE);
            b.setStyle("-fx-border-color:black");
            HBox.setHgrow(b, Priority.ALWAYS);

            // 스타일 적용. 자리마다 다른 색
            if (i < 6) {
                b.setStyle("-fx-background-color:lightblue");
                VBox.setMargin(hbox, new Insets(0, 0, 10, 0));
            } else {
                b.setStyle("-fx-background-color:lightpink");
                VBox.setMargin(hbox, new Insets(10, 0, 10, 0));
            }
```

좌석 선택 노트르담드파리 2024-01-17 23:00:00

뒤로가기

Enterpark ticket

원하시는 좌석을 선택해주세요

STAGE

선택됐던 좌석

A-1	A-2	A-3	A-4	A-5	A-6	A-7	A-8	A-9	A-10
B-1	B-2	B-3	B-4	B-5	B-6	B-7	B-8	B-9	B-10
C-1	C-2	C-3	C-4	C-5	C-6	C-7	C-8	C-9	C-10
D-1	D-2	D-3	D-4	D-5	D-6	D-7	D-8	D-9	D-10
E-1	E-2	E-3	E-4	E-5	E-6	E-7	E-8	E-9	E-10
F-1	F-2	F-3	F-4	F-5	F-6	F-7	F-8	F-9	F-10
G-1	G-2	G-3	G-4	G-5	G-6	G-7	G-8	G-9	G-10
H-1	H-2	H-3	H-4	H-5	H-6	H-7	H-8	H-9	H-10
I-1	I-2	I-3	I-4	I-5	I-6	I-7	I-8	I-9	I-10
J-1	J-2	J-3	J-4	J-5	J-6	J-7	J-8	J-9	J-10

좌석등급	가격
VIP 좌석	100,000원
일반좌석	30,000원

선택된 좌석

VIP 좌석 C-6 10만원

좌석 선택 완료

결제페이지 이동

```
btnBack.setOnAction((e)->{
    selectDate = null;
    selectTime = null;
    selectMusical = null;
    alertWarning("좌석 선택을 취소하였습니다.");
    Stage end = (Stage)btnBack.getScene().getWindow();
    end.close();
});
```

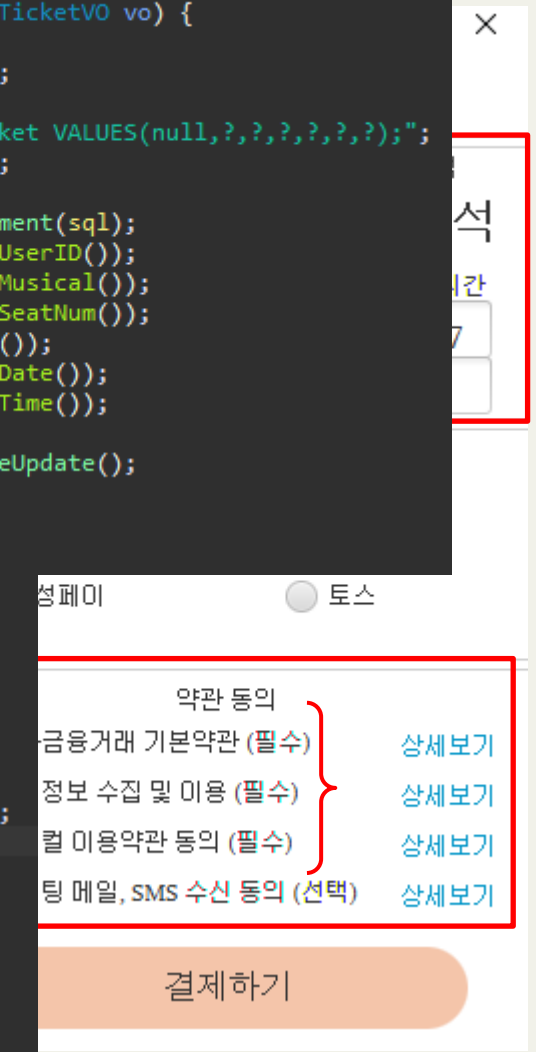
```
public void handle(ActionEvent arg0) {
    reservSeat = b.getText();
    selectS.setText(reservSeat);
    String receiveData = reservSeat;
    if (selectedButton != null && btnStyle != null) {
        selectedButton.setStyle(btnStyle);
    }
    selectedButton = b;
    btnStyle = b.getStyle();
    b.setStyle("-fx-background-color:gray");
    b.setDefaultButton(false);
    String[] ticket = receiveData.split("\\-");
    String code = ticket[0];
```

```
}else if(order.equals("2")) {
    // 2|0|data...
    // 2|0|33,레베카,E-8,100000,2024-01-11,18:00:00
    // 결제 관련 요청 처리에 대한 서버의 결과
    if(datas[1].equals("0")) {
        String[] ticket = datas[2].split(",");
        //  0   1      2  ...
        // [33][레베카][
        TicketVO vo =
        vo.setUserID(t
        vo.setMusical(
        vo.setSeatNum(
        vo.setPay(Inte
        vo.setDate(tic
        vo.setTime(tic
        System.out.println(vo);
        boolean isReservation = ticketDAO.reservationTicket(vo);
        sendData("2|0|"+isReservation);
```

ticketNum	userID	musical	seatNum	pay	Date	time
141	33	레미제라블	B-6	100000	2024-01-18	22:00:00
142	33	노트르담드파리	D-7	100000	2024-01-18	23:00:00
143	33	노트르담드파리	B-5	100000	2024-01-18	23:00:00
144	33	노트르담드파리	F-5	30000	2024-01-18	23:00:00
145	0	노트르담드파리	D-6	100000	2024-01-17	23:00:00

```
// 로그인 했던 정보를 바탕으로 예매한 정보를 mysql에 입력
@Override
public boolean reservationTicket(TicketVO vo) {

    boolean isReservation = false;

    String sql = "INSERT INTO ticket VALUES(null,?,?,?,?,?,?,?);";
                                                    ction();

                                                    reStatement(sql);
                                                    vo.getUserID());
                                                    vo.getMusical());
                                                    vo.getSeatNum());
                                                    .getPay());
                                                    pstmt.setString(5, vo.getDate());
                                                    pstmt.setString(6, vo.getTime());

                                                    int result = pstmt.executeUpdate();
                                                    if(result == 1) {
                                                        // 예약 정상 등록
                                                        isReservation = true;
```

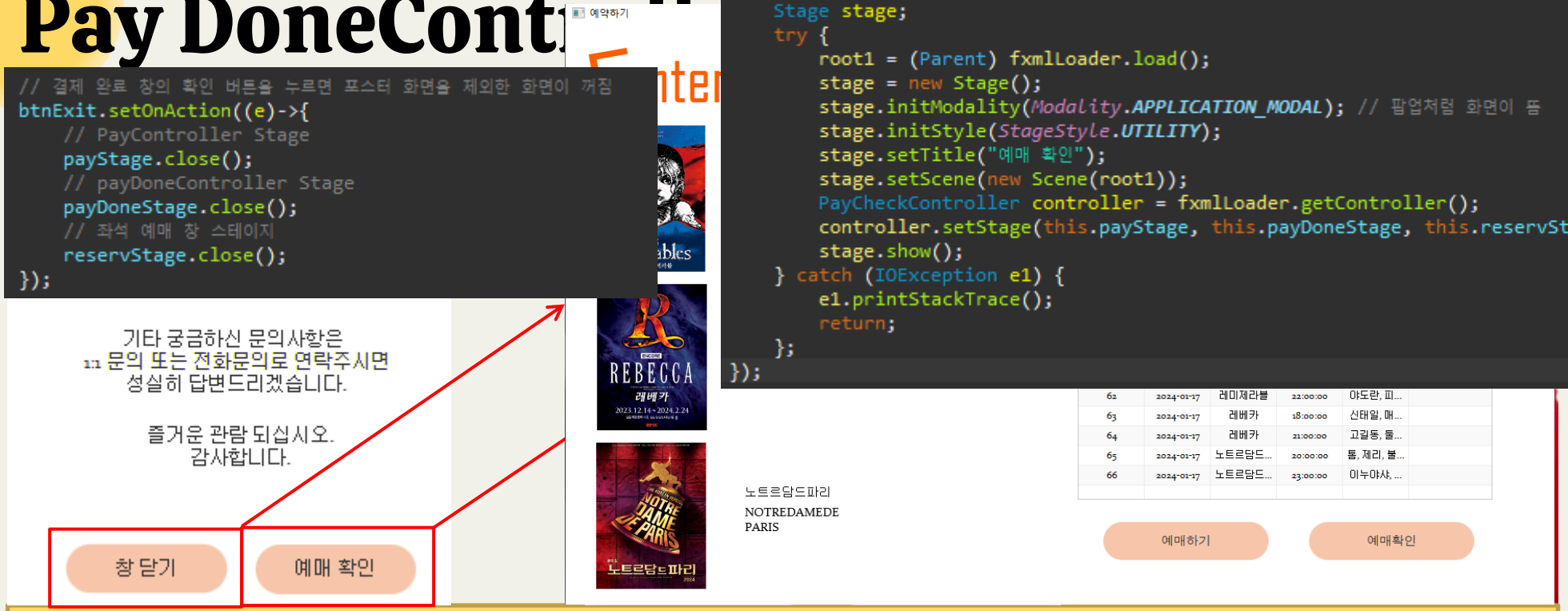
```
// 서버에서 데이터 받아오기
// 데이터가 불러와지지 않으면 결제 완료 창이 뜨지 않음
@Override
public void receiveData(String message) {
    System.out.println("receive pay : " + message);
    String paying = message.split("\\|")[2];
    // 2|0|true, 2|0|false
    if (Boolean.parseBoolean(paying)) {
        Platform.runLater(() -> {
            try {
                FXMLLoader fxmlLoader = new FXMLLoader(getClass().getResource("/pay/PayDone.fxml"));

                Parent root1 = fxmlLoader.load();
                Stage stage = new Stage();
                stage.initModality(Modality.APPLICATION_MODAL); // 팝업처럼 화면이 뜸
                stage.initStyle(StageStyle.UNDECORATED);
                stage.setTitle(Main.reservTicket.getMusical()+" 결제 완료");
                stage.setScene(new Scene(root1));
                PayDoneController controller = fxmlLoader.getController();
                controller.setStage(this.stage, stage, this.reservStage);
                stage.show();
            } catch (IOException e1) {
                e1.printStackTrace();
            }
        });
```



Pay DoneCont

```
// 결제 완료 창의 확인 버튼을 누르면 포스터 화면을 제외한 화면이 꺼짐
btnExit.setOnAction((e)->{
    // PayController Stage
    payStage.close();
    // payDoneController Stage
    payDoneStage.close();
    // 좌석 예매 창 스테이지
    reservStage.close();
});
```

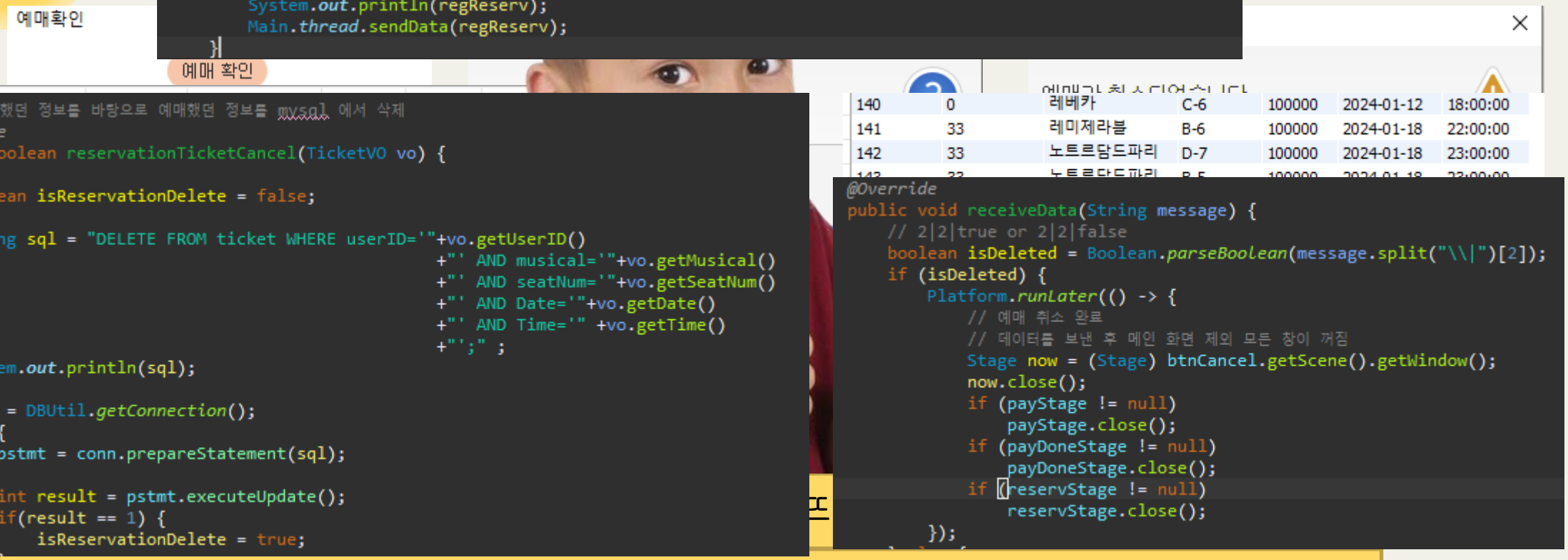
```
// 결제 완료 창에서 예매 확인 버튼을 누르면 PayCheck 으로 넘어감
btnCheck.setOnAction((e)->{
    FXMLLoader fxmlLoader = new FXMLLoader(getClass().getResource("/pay/PayCheck.fxml"));
    Parent root1;
    Stage stage;
    try {
        root1 = (Parent) fxmlLoader.load();
        stage = new Stage();
        stage.initModality(Modality.APPLICATION_MODAL); // 팝업처럼 화면이 뜸
        stage.initStyle(StageStyle.UTILITY);
        stage.setTitle("예매 확인");
        stage.setScene(new Scene(root1));
        PayCheckController controller = fxmlLoader.getController();
        controller.setStage(this.payStage, this.payDoneStage, this.reservStage);
        stage.show();
    } catch (IOException e1) {
        e1.printStackTrace();
        return;
    };
});
```



1. 결제를 성공적으로 완료하게 되면 창을 닫거나, 필요에 따라서 예매 확인을 할 수 있습니다.

2. 창을 닫으면 다시 포스터 화면으로, 예매 확인을 누르면 예매한 내용을 확인 할 수 있습니다.

```java
// 예매 취소를 진행 할 경우 알림이 뜨며 예매 취소가 됨.
if (result.get() == ButtonType.OK) {
    System.out.println("예매 취소");
    Alert alert1 = new Alert(Alert.AlertType.WARNING);
    alert1.setTitle("예매 취소 확인");
    alert1.setHeaderText("예매가 취소되었습니다.");
    alert1.setContentText("다음에 다시 만나요 ㅠㅠ");
    Optional<ButtonType> result1 = alert1.showAndWait();
    if (result1.get() == ButtonType.OK) {
        // 예약 취소 화면에서 예매했던 구매자의 데이터를 보냄
        // 데이터를 보낸 후 데이터 삭제
        TicketVO vo = Main.reservTicket;
        String regReserv = "2|2|" + vo.getUserID() + "," + vo.getMusical() + "," + vo.getSeatNum() + ","
                + vo.getPay() + "," + vo.getDate() + "," + vo.getTime();
        System.out.println(regReserv);
        Main.thread.sendData(regReserv);
    }
```




```java
// 로그인 했던 정보를 바탕으로 예매했던 정보를 mysql 에서 삭제
@Override
public boolean reservationTicketCancel(TicketVO vo) {

    boolean isReservationDelete = false;

    String sql = "DELETE FROM ticket WHERE userID='"+vo.getUserID()
                                          +"' AND musical='"+vo.getMusical()
                                          +"' AND seatNum='"+vo.getSeatNum()
                                          +"' AND Date='"+vo.getDate()
                                          +"' AND Time='" +vo.getTime()
                                          +"';" ;

    System.out.println(sql);

    conn = DBUtil.getConnection();
    try {
        pstmt = conn.prepareStatement(sql);

        int result = pstmt.executeUpdate();
        if(result == 1) {
            isReservationDelete = true;
```

140	0	레베카	C-6	100000	2024-01-12	18:00:00
141	33	레미제라블	B-6	100000	2024-01-18	22:00:00
142	33	노트르담드파리	D-7	100000	2024-01-18	23:00:00

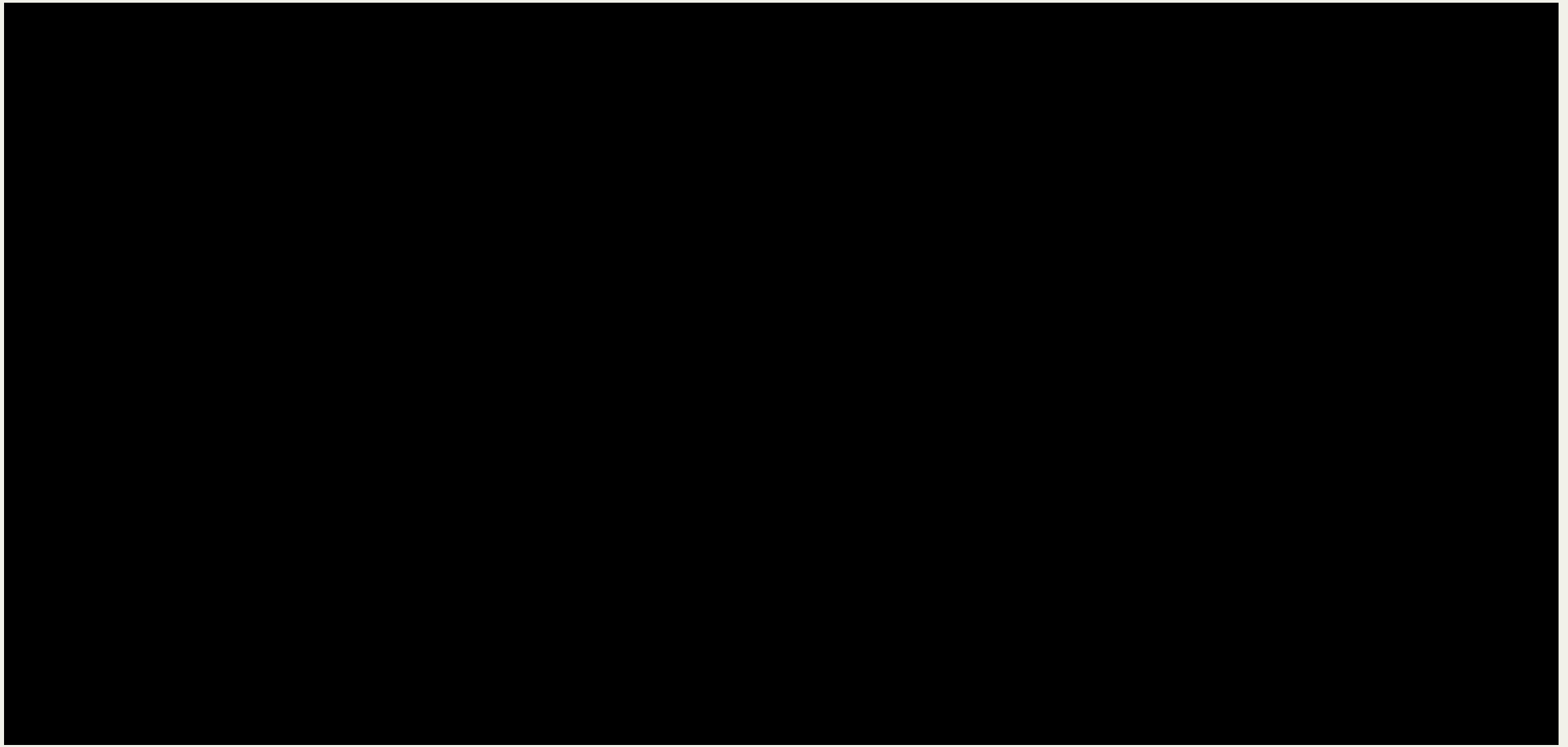
```java
@Override
public void receiveData(String message) {
    // 2|2|true or 2|2|false
    boolean isDeleted = Boolean.parseBoolean(message.split("\\|")[2]);
    if (isDeleted) {
        Platform.runLater(() -> {
            // 예매 취소 완료
            // 데이터를 보낸 후 메인 화면 제외 모든 창이 꺼짐
            Stage now = (Stage) btnCancel.getScene().getWindow();
            now.close();
            if (payStage != null)
                payStage.close();
            if (payDoneStage != null)
                payDoneStage.close();
            if (reservStage != null)
                reservStage.close();
        });
```

2. 취소 화면에서 확인을 누르면 예매자의 정보를 서버로 보내고, 그 데이터를 삭제합니다.

04

After that we will talk about this.

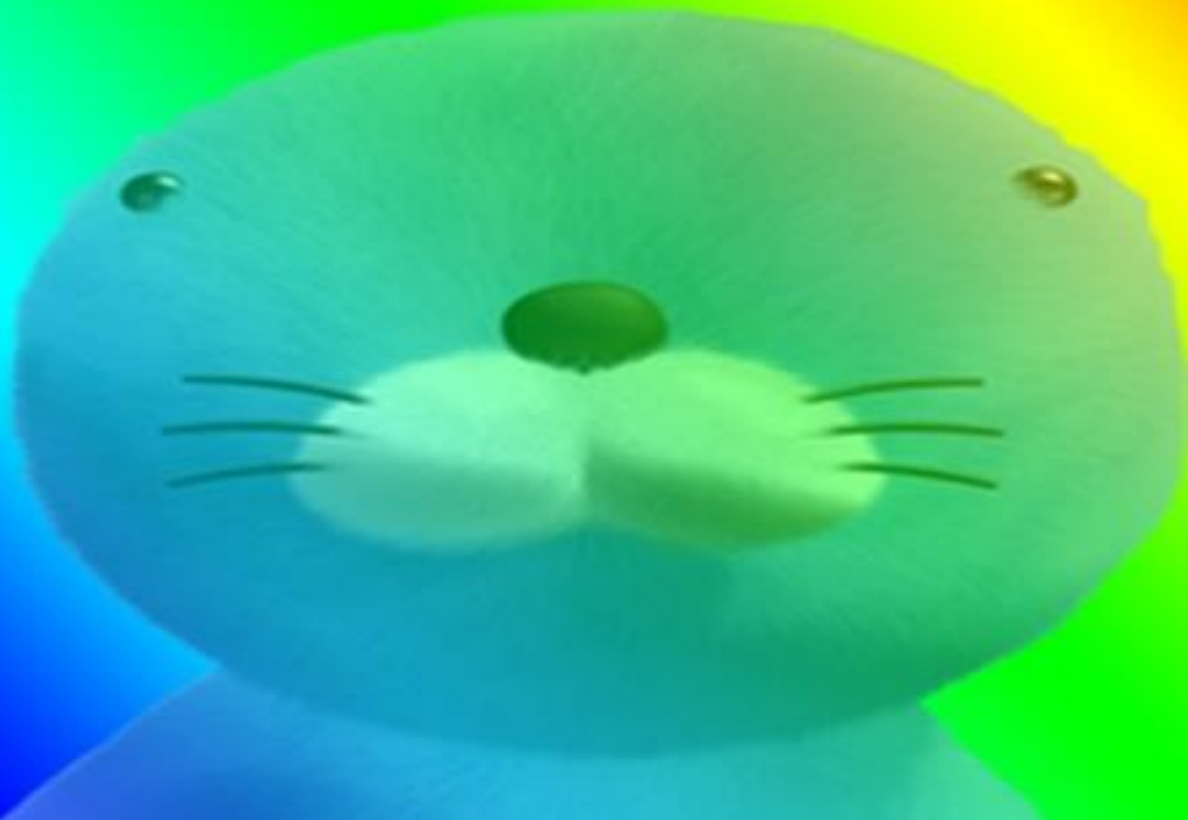
수행결과 / 개선사항



개선사항

- 로그아웃 및 회원정보 수정
- 뮤지컬 선택시 해당 뮤지컬 예매 정보만 출력
- 다인 티켓예매 가능
- 현재 날짜 이전 예매 취소 불가
- 관리자와의 채팅

질문 있으십니까?



Thank you!