



# 주 차 관 리 시 스 템

데 이 터 베 이 스   및   실 습

B289024 김태희  
B289051 성에린  
B489027 박영빈

# Contents

01 / 제안 이유

02 / 요구 사항

03 / 주요 기능

04 / E/R 다이어그램

05/ 최종 결과

# 1 제안 이유

# 제안 이유

---

- 주차 시 시간 낭비를 최소화
- 주차공간을 좀 더 효율적으로 활용
- 고객을 체계적으로 효율적 관리

## 2 요구 사항

## 요 구 사 항

---

- 주차 관리 시스템의 규모는 차량을 최대 90대까지 관리
- 주차장을 이용하는 모든 고객의 주차번호, 차량번호, 입/출차 시간, 이용요금 정보 기록
- 고객이 출차할 경우, 요금은 시간에 따라 자동으로 계산 후 출력

# 3 주요 기능

# 주요기능

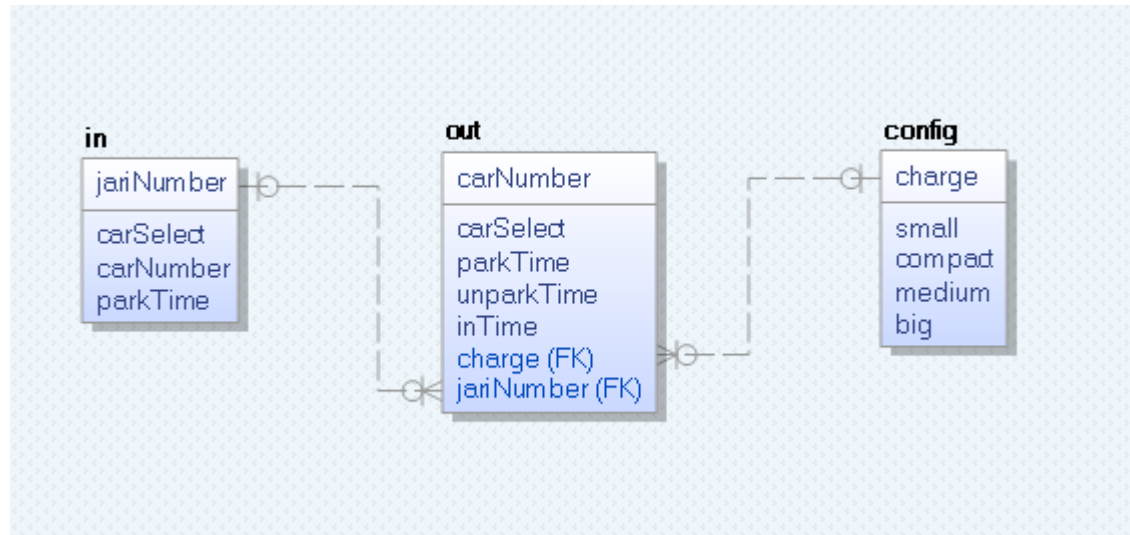
---

1. **입출차 기록 기능**: 주차장을 이용하는 고객에 대해 아래의 정보를 입력 받아 데이터베이스에 저장  
→ 주차 번호, 차량 번호, 입차 시간, 출차 시간
2. **이용 요금 계산 기능**: 고객이 출차할 경우, 해당 고객의 출차 시간에서 입차 시간을 뺀 이용 시간을 요금표에 대입하여 그에 해당하는 이용 요금을 출력



# 4 E/R 다이어그램

# E/R 다이어그램



# 5 최종 결과

# 최 종 결 과

주차 관리 시스템 (db2\_7 칠면조)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40	41	42	43	44	45
46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70	71	72	73	74	75
76	77	78	79	80	81	82	83	84	85	86	87	88	89	90

내역보기

요금설정

메인화면

# 최 종 결 과

```

7 @SuppressWarnings("serial")
8 class mainWindow extends JFrame implements Runnable, ActionListener {
9     JTextField blank; // 상단의 시계표시를 위해 공백을 만드는 텍스트필드
10    public JButton Btn[] = new JButton[150]; // 자리 버튼
11    JTextArea condition; // 상황판 텍스트 에어리어
12    JButton configBtn; // 요금설정 버튼
13    String currentDate = ""; // 현재 날짜를 저장
14    String currentTime = ""; // 현재 시간을 저장
15    JButton history; // 내역보기 버튼
16    int jariNumber; // 자리번호
17    JButton notUseBtn[] = new JButton[100]; // 사용하지 않는 버튼(자리 사이의 버튼)
18    readFile obj = new readFile(); // 파일을 읽어들임
19    JPanel pan2; // 패널2
20    JPanel pan3; // 패널3
21
22    // 생성자
23    public mainWindow() {
24        setTitle("주차 관리 시스템 (db2_7 칠면조)"); // 제목표시줄
25        setSize(1000, 700); // 사이즈
26        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); // 종료버튼 클릭시 종료함
27        //pan1(); // 메소드 호출
28        pan2(); // 메소드 호출
29        pan3(); // 메소드 호출
30        gridInit(); // 메소드 호출
31        setVisible(true); // 창을 보여줌
32        setResizable(false); // 크기변경 불가
33    }

```

# 최 종 결 과

```

@SuppressWarnings("deprecation")
// 버튼 이벤트
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == configBtn) { // 요금설정 버튼
        new config(); // 요금설정 클래스 실행
    } else if (e.getSource() == history) { // 내역보기 버튼
        new logWindow(); // 내역보기 클래스 실행
    } else if (Integer.parseInt(e.getActionCommand()) >= 0 // 버튼의 이름을 정수형태로 변환하여
        && Integer.parseInt(e.getActionCommand()) <= Btn.length) { // 0~150사이의 숫자면
        jariNumber = Integer.parseInt(e.getActionCommand()); // 자리번호 변수에 저장하고
        new parkWindow(jariNumber); // 그것을 매개변수로 입차 창 띄우기
        dispose(); // 창을 닫음
    } else { // 0~150사이의 숫자가 아닐시에는
        for (int i = 1; i < Btn.length; i++) { // 버튼을 처음부터 끝까지 검색
            String temp = Btn[i].getLabel(); // temp에 버튼의 라벨을 저장
            int temp2 = Integer.parseInt(temp); // temp1에 저장한 라벨을 정수형태로 변환하여 저장해 둠
            if (temp2 == Integer.parseInt(e.getActionCommand())) {
                // temp2에 저장해둔 라벨과 액션이벤트로 들어온 매개변수가 같다면
                // 자가 주차되어 있는 자리가 바로 그 자리 이므로
                jariNumber = i; // 자리번호는 i
                break; // 루프를 나옴
            }
        }
        new unparkWindow(jariNumber); // 자리번호를 매개변수로 출차 창 띄움
        dispose(); // 창을 닫음
    }
}

@SuppressWarnings("deprecation")
// 버튼을 생성하는 메소드
public void gridInit() {

```

# 최 종 결 과

```
// 버튼을 생성하는 메소드
public void gridInit() {
    for (int i = 1; i <= 15; i++) {
        pan2.add(Btn[i] = new JButton(i + ""));
        Btn[i].addActionListener(this);
    }
    for (int i = 1; i <= 15; i++) {
        pan2.add(notUseBtn[i] = new JButton(""));
        notUseBtn[i].setEnabled(false);
    }
    for (int i = 16; i <= 45; i++) {
        pan2.add(Btn[i] = new JButton(i + ""));
        Btn[i].addActionListener(this);
    }
    for (int i = 16; i <= 30; i++) {
        pan2.add(notUseBtn[i] = new JButton(""));
        notUseBtn[i].setEnabled(false);
    }
    for (int i = 46; i <= 75; i++) {
        pan2.add(Btn[i] = new JButton(i + ""));
        Btn[i].addActionListener(this);
    }
    for (int i = 31; i <= 45; i++) {
        pan2.add(notUseBtn[i] = new JButton(""));
        notUseBtn[i].setEnabled(false);
    }
    for (int i = 76; i <= 90; i++) {
        pan2.add(Btn[i] = new JButton(i + ""));
        Btn[i].addActionListener(this);
    }
    // 여기까지 버튼 생성
    int i = 0;
    // 차량이 주차되어 있는 자리의 라벨과 색깔을 바꿈
    while (i < obj.length) { // 객체의 length 변수만큼 loop
        int temp = Integer.parseInt(obj.jariNumber[i]); // 파일에 저장된 자리번호를
        // temp에 정수형태로 저장
        Btn[temp].setLabel(obj.carNumber[i]); // 자리번호의 버튼의 라벨을 차량번호로 바꿈
        Btn[temp].setForeground(new Color(255, 0, 0)); // 색깔을 붉은색으로 바꿈
        i++;
    }
}
```

# 최 종 결 과

```

105 // 중앙 버튼부분 패널
106
107 public void pan2() {
108     pan2 = new JPanel(); // 패널 생성
109     GridLayout layout = new GridLayout(9, 1); // 그리드 레이아웃 생성
110     pan2.setLayout(layout); // 레이아웃 설정
111     add(pan2, "Center"); // 패널2 부착
112 }
113
114 // 하단 상황판 부분 패널
115 public void pan3() {
116     pan3 = new JPanel(); // 패널 생성
117     pan3.add(condition = new JTextArea()); // 상황판을 생성
118     for (int i = 0; i < 11; i++) { // 상황판에 message 클래스에 저장된 메시지를 출력함
119         condition.append(message.message[i]);
120     }
121     condition.setPreferredSize(new Dimension(700, 200)); // 상황판 사이즈
122     condition.setEditable(false); // 편집 불가
123
124     condition.setPreferredSize(new Dimension(700, 200)); // 상황판 사이즈
125     condition.setEditable(false); // 편집 불가
126     pan3.add(history = new JButton("내역보기")); // 버튼 부착
127     pan3.add(configBtn = new JButton("요금설정")); // 버튼 부착
128     history.addActionListener(this); // 액션리스너 추가
129     configBtn.addActionListener(this); // 액션리스터 추가
130     add(pan3, "South"); // 패널3 부착
131 }
132
133 // 스레드 실행 부분
134 public void run() {
135     while (true) { // 반복문
136         Date d = new Date(); // 날짜 객체 생성
137         SimpleDateFormat date = new SimpleDateFormat("yyyy년 MM월 dd일"); // 형식 지정
138         SimpleDateFormat time = new SimpleDateFormat("hh시 mm분 ss초"); // 형식 지정
139         currentDate = date.format(d); // 현재 날짜를 위에 저장해둔 형식으로 문자열 저장
140         currentTime = time.format(d); // 현재 시간을 위에 저장해둔 형식으로 문자열 저장
141         try {
142             Thread.sleep(1000); // 0.5초 틈을 두고
143             repaint(); // 다시그리기를 반복
144         } catch (InterruptedException e) {
145         }
146     }
147 }

```



# 최 종 결 과

```
// 입차 부분
StringBuilder builder = new StringBuilder();
builder.append(jariNumber);//
builder.append(",");//
builder.append(carSelect);//
builder.append(",");//
builder.append(carNumber);//
builder.append(",");//
builder.append(parkTime);//
DataBase.execute(DataBase.queryBuilder("Insert into in values(",builder.toString(),")"));
```

입차화면

# 최 종 결 과

주차 관리 시스템 (db2\_7 칠면조)

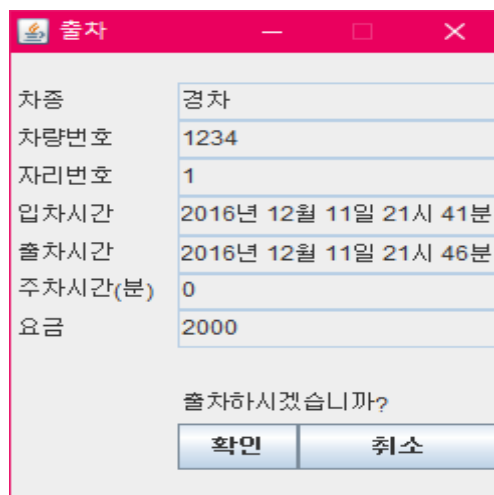
1234	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40	41	42	43	44	45
46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70	71	72	73	74	75
76	77	78	79	80	81	82	83	84	85	86	87	88	89	90

1번 자리로 1234번 차량이 2016년 12월 11일 21시 41분에 입차되었습니다.

내역보기    요금설정

입차결과

# 최 종 결 과



차종	경차
차량번호	1234
자리번호	1
입차시간	2016년 12월 11일 21시 41분
출차시간	2016년 12월 11일 21시 46분
주차시간(분)	0
요금	2000

출차하시겠습니까?

```

StringBuilder builder = new StringBuilder();
builder.append(jariNumber);//
builder.append(",");//
builder.append(carSelect);//
builder.append(",");//
builder.append(carNumber);//
builder.append(",");//
builder.append(parkTime);//
builder.append(",");//
builder.append(unparkTime);//
builder.append(",");//
builder.append(inTime);//
builder.append(",");//
builder.append(charge);//
DataBase.executeUpdate(DataBase.queryBuilder("insert into out values('",builder.toString(),"')"));
  
```

출차화면

# 최 종 결 과

주차 관리 시스템 (db2\_7 칠면조)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40	41	42	43	44	45
46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70	71	72	73	74	75
76	77	78	79	80	81	82	83	84	85	86	87	88	89	90

1번 자리로 1234번 차량이 2016년 12월 11일 21시 41분에 입차되었습니다.  
1번 자리에서 1234번 차량이 2016년 12월 11일 21시 46분에 출차되었습니다.

내역보기    요금설정

## 출차결과

# 최 종 결 과

	경차	소형차	중형차	대형차
1시간 기본요금	2000	3000	4000	5000
이후 30분당 요금	500	1000	1500	2000
최대요금	10000	15000	15000	20000
	확인	취소		

```
// 액션리스너
public void actionPerformed(ActionEvent ae) {
    if (ae.getSource() == confirm) {// 확인 버튼
        // temp에 텍스트필드에서 값을 얻어와서 저장함
        String temp = tf0.getText() + "," + tf1.getText() + ","
            + tf2.getText() + "," + tf3.getText() + "," + tf4.getText()
            + "," + tf5.getText() + "," + tf6.getText() + ","
            + tf7.getText() + "," + tf8.getText() + "," + tf9.getText()
            + "," + tf10.getText() + "," + tf11.getText();
        // 파일을 열어서 값을 저장함
        DataBase.executeUpdate("delete from config where name='config'");
        DataBase.executeUpdate(DataBase.queryBuilder("insert into config values('charge', '",temp,"')"));
        dispose();// 창 닫음
    } else if (ae.getSource() == cancel) {// 취소버튼
        dispose();// 창닫음
    }
}
```

요금테이블



Q & A



Thank you