

# JUST HAB' IT

3조 커밋이 뭐였조  
차진서, 이해승, 채도혁, 김해인

# CONTENTS

---

01  
역할 및 프로젝트  
소개

02  
개발 환경 및  
개발 일정

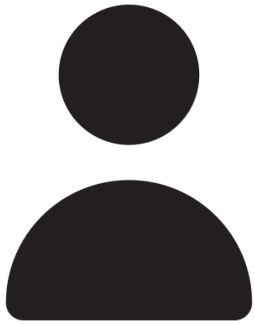
03  
UI 설계도

04  
시연

01

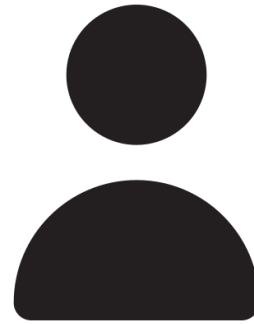
역할 및 프로젝트 소개

## 01. 팀원 및 역할 소개



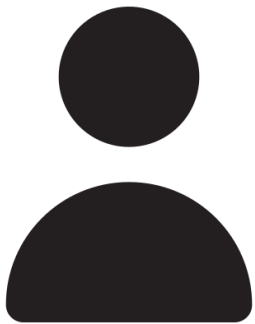
### 차진서

- 로그인, 회원가입, 회원탈퇴, 정보수정  
GUI 설계 및 기능 구현
- 데이터베이스 설계
- 일정 조절



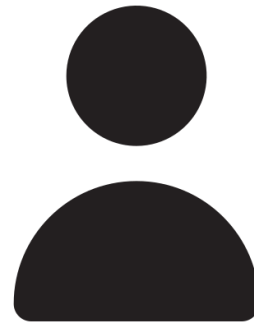
### 이해승

- 습관기록 GUI 설계 및 기능 구현
- 화면 간 이동 구현
- 기본 디자인 설계



### 채도혁

- 습관추가 기능 구현



### 김해인

- 프로젝트 기획
- 습관조회, 습관삭제, 메인페이지  
GUI 설계 및 기능 구현

## 01. 프로젝트 소개

# JUST HAB' IT

사용자에게 맞는 최적의 습관 기록 어플리케이션

<유연한 습관 기록 방식>  
타이머, 횟수, 실행 요일을 선택 가능

<직관적 습관 성취도 시각화>  
성취 여부에 따라 달력의 색깔 변화,  
전체 습관 성취도에 따른 레벨 변화

## 01. 프로젝트 소개

로그인  
회원가입

메인  
페이지

습관추가

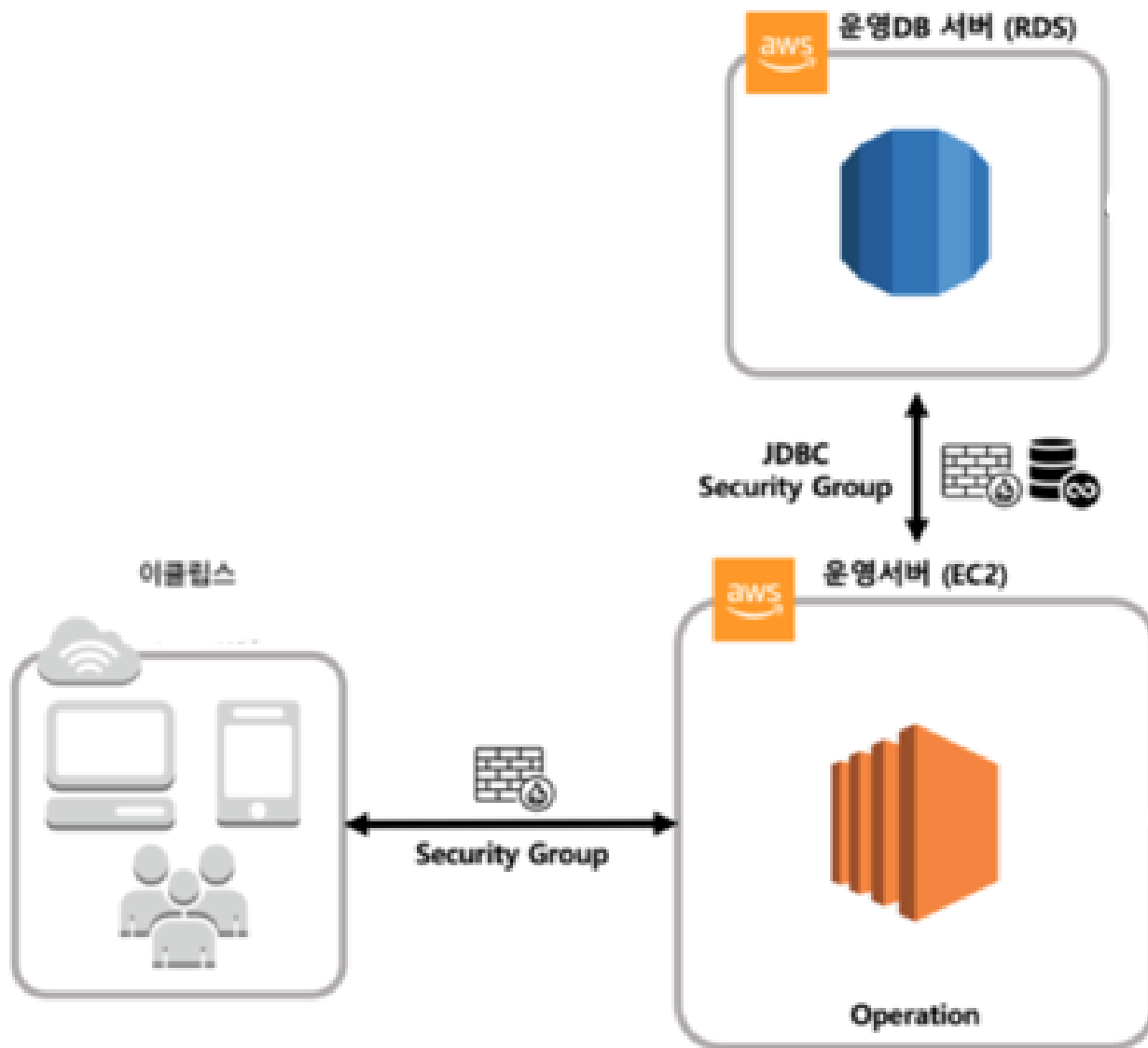
습관기록

마이  
페이지

# 02

개발 환경 및 개발 일정

## 02. 개발 환경 및 사용언어





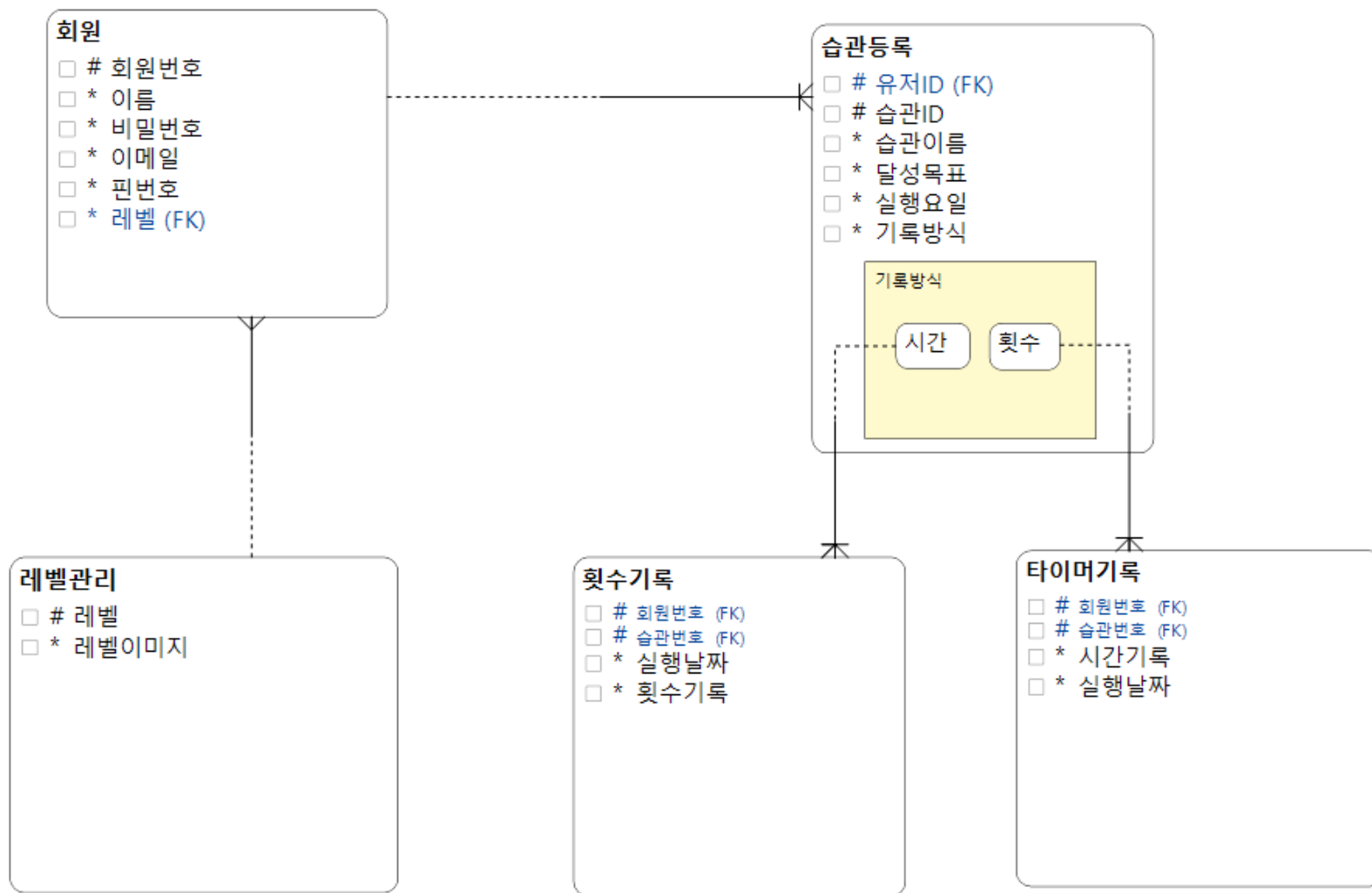
## 02. 개발 일정

	03.15(월) ~ 03.29(월)	03.30(금) ~ 04.05(월)	04.06(화) ~ 04.14(수)	04.15(목)
프로젝트 기획 회의 및 보고서				
화면 UI 디자인 회의				
프로젝트 구현/ 디버깅 / 테스트				
프로젝트 발표 및 시연				

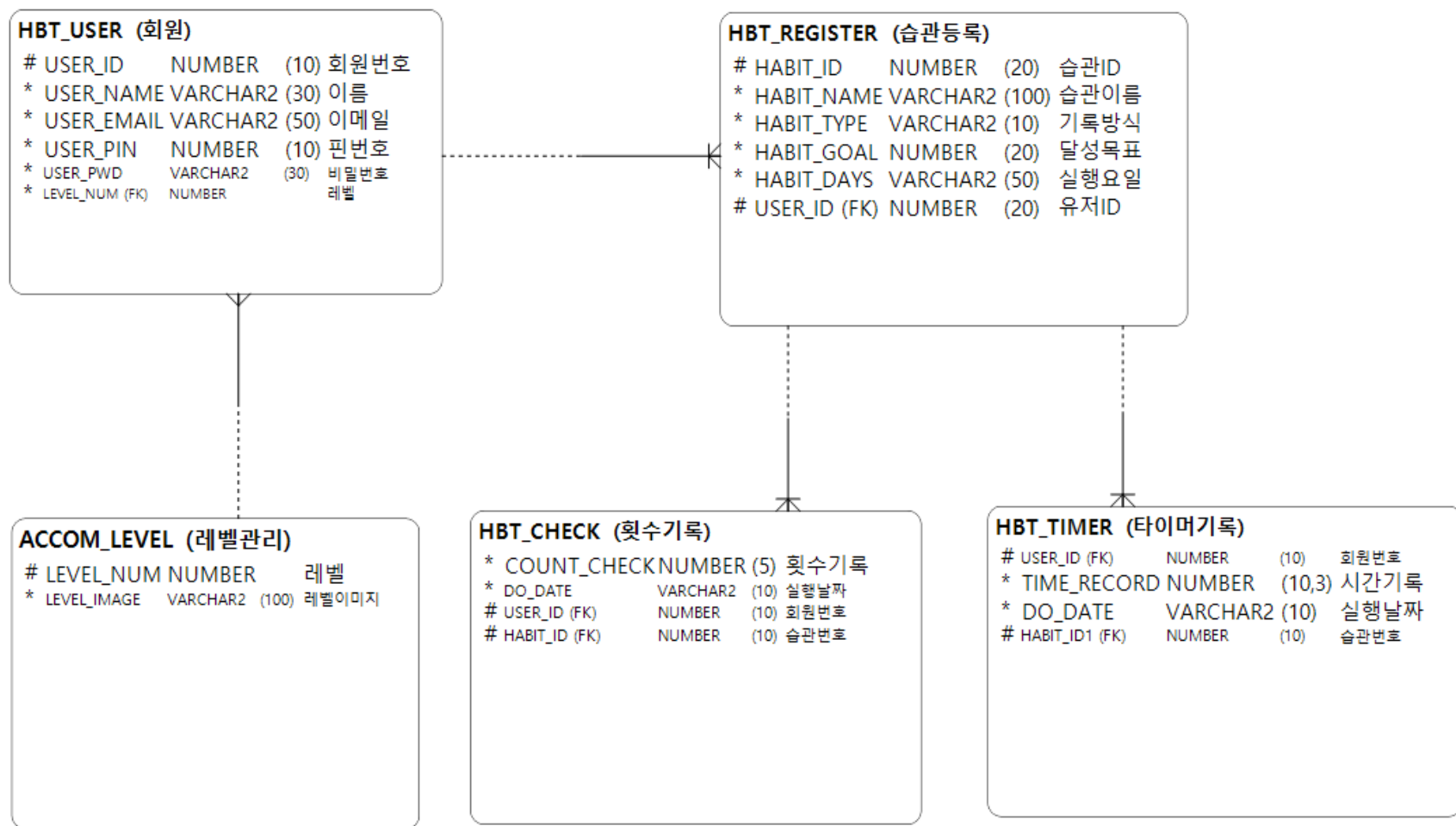
# 03

UI 설계도

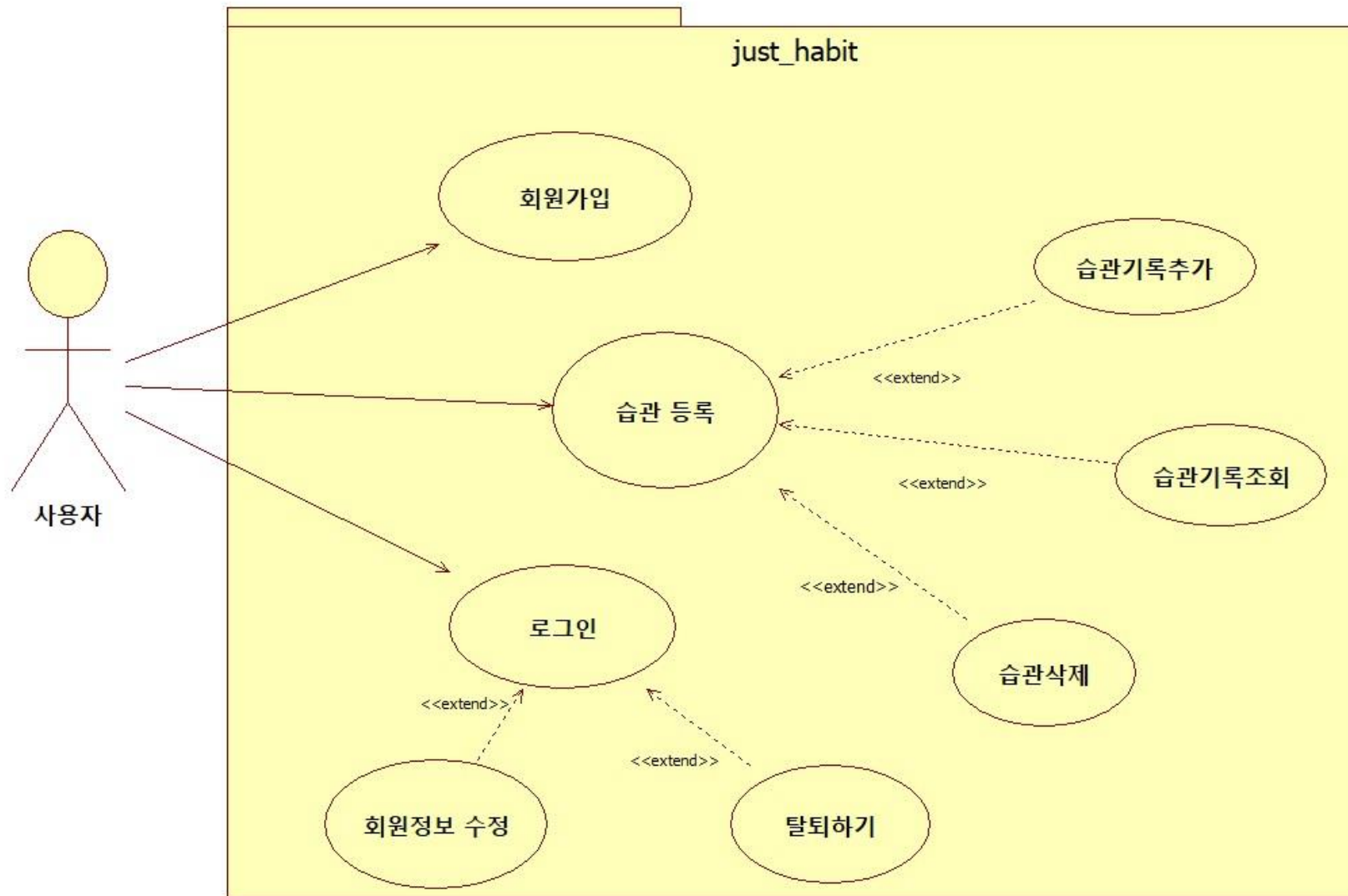
## 03-1. 논리 모델



## 03-2. 물리 모델



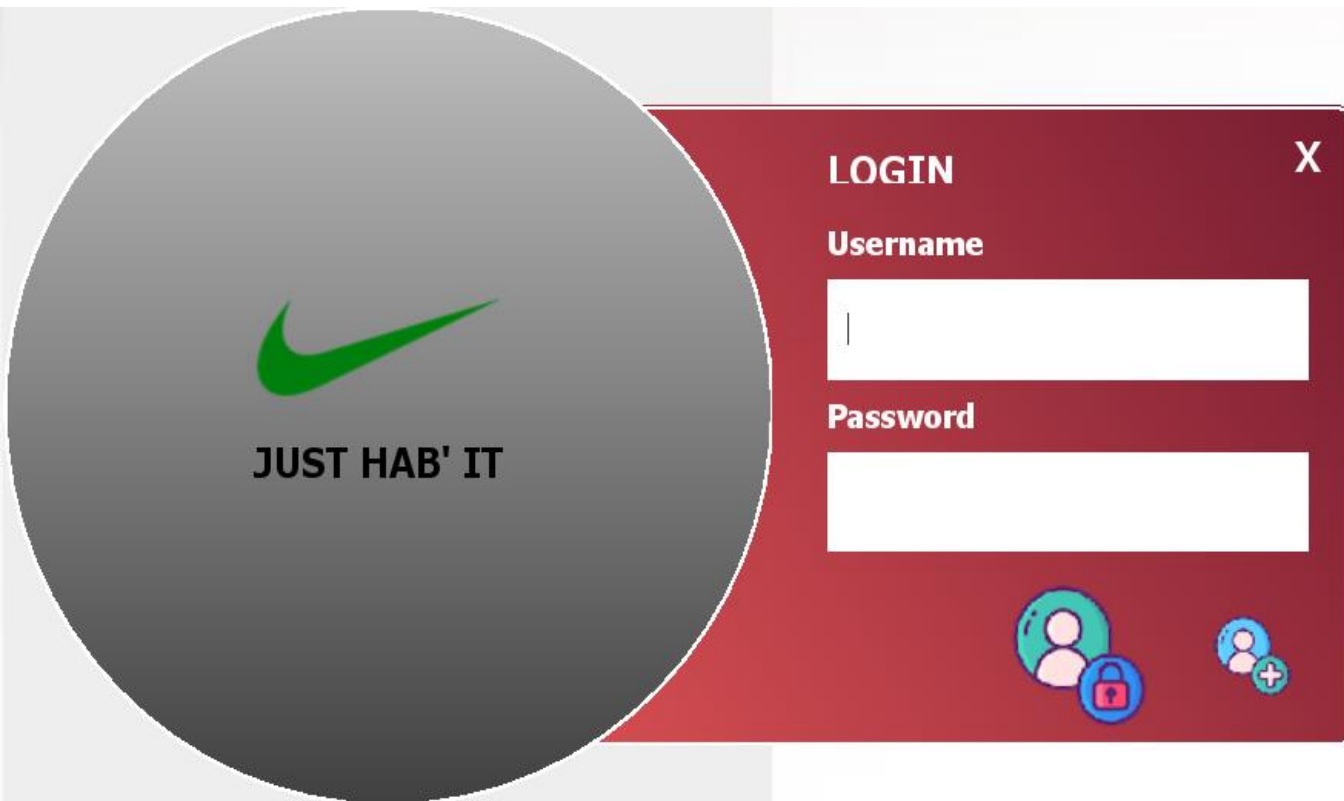
### 03. 유스케이스 다이어그램



04

시연

## 04-1. 로그인



## Controller

```
public boolean loginCheck(String loginId, String loginPwd) {  
  
    /* 유저가 입력한 로그인 아이디/패스워드 유효성 체크 */  
  
    // 1. 입력한 정보가 공백으로 이루어진 경우 로그인 실패  
    if(loginId.trim().equals("") || loginPwd.trim().equals("")) {  
        return false;  
    }  
  
    return userService.login(loginId, loginPwd);  
}
```

## Service

```
public boolean login(String loginId, String loginPwd) {  
  
    Connection con = getConnection();  
  
    boolean IsloggedIn = userDAO.selectUser(con, loginId, loginPwd );  
  
    close(con);  
  
    return IsloggedIn;  
}
```

## 04-1. 로그인



## Query

```
<entry key="login">
    SELECT
        user_id
    FROM admin.hbt_user
    WHERE user_name = ?
    AND user_pwd = ?
</entry>
```

## DAO

```
public boolean selectUser(Connection con, String loginId, String loginPwd) {

    PreparedStatement psmt = null;
    ResultSet rset = null;

    String query = prop.getProperty("login");

    int result = 0;

    try {
        psmt = con.prepareStatement(query);
        psmt.setString(1, loginId);
        psmt.setString(2, loginPwd);

        rset = psmt.executeQuery();

        if(rset.next()) {
            result = rset.getInt(1);
            FirstFrame.loggedUserID = result;
        }
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        close(rset);
        close(psmt);
    }

    return result == 0 ? false: true;
}
```



## 04-1. 회원가입



```

suLabel.addMouseListener(new MouseAdapter() {
    @Override
    public void mouseClicked(MouseEvent e){
        String result = userController.signupCheck(sign_username.getText().toLowerCase()
            , new String(sign_pwd.getPassword()), new String(sign_pwd_check.getPassword())
            , sign_email.getText(), sign_PIN.getText());

        if(result.equals("회원가입이 성공적으로 완료되었습니다 :")) {
            JOptionPane.showMessageDialog(signup, result + "\n로그인 화면으로 이동합니다.");
            frame.setVisible(false);
            FirstFrame.main(null);
        }
        else {
            JOptionPane.showMessageDialog(signup, result);
        }
    }
});
    
```

## 04-1. 회원가입

```
public String signupCheck(String signup_name, String pwd1, String pwd2,
    String email, String pin) {

    /* 유저가 입력한 회원가입용 데이터 유효성 체크 */

    //1. 모든 입력값 공백으로 이루어졌는지 확인
    if(signup_name.trim().equals("") || pwd1.trim().equals("") ||
        pwd2.trim().equals("") || email.trim().equals("") ||
        pin.trim().equals(""))
        return "모든 필요한 정보가 채워지지 않았습니다";

    //2. 입력한 두번의 비밀번호가 일치하는지 확인.
    if(!pwd1.equals(pwd2))
        return "입력하신 두 비밀번호가 일치하지 않습니다";

    //3. 입력한 아이디가 이미 존재하는지 확인
    if(!userService.userIdCheck(signup_name))
        return "이미 존재하는 유저입니다";

    /* UserDTO를 생성 */
    UserDTO registerUser = new UserDTO();
    registerUser.setUserName(signup_name);
    registerUser.setUserPwd(pwd1);
    registerUser.setUserEmail(email);
    registerUser.setUserPin(Integer.parseInt(pin));

    /* 회원가입 진행 */
    boolean isRegistered = userService.register(registerUser);

    return isRegistered == true ? "회원가입이 성공적으로 완료되었습니다 :)" : "에러발생";
}
```

```
public boolean checkID(Connection con, String signup_name) {
    PreparedStatement psmt = null;
    ResultSet rset = null;

    boolean available = true;

    String query = prop.getProperty("checkUsername");

    try {
        psmt = con.prepareStatement(query);
        psmt.setString(1, signup_name);

        rset = psmt.executeQuery();

        if(rset.next())
            available = false;
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        close(rset);
        close(psmt);
    }

    return available;
}

public boolean registerUser(Connection con, UserDTO registerUser) {
    PreparedStatement psmt = null;

    int result = 0;

    String query = prop.getProperty("registerUser");

    try {
        psmt = con.prepareStatement(query);
        psmt.setString(1, registerUser.getUserName());
        psmt.setString(2, registerUser.getUserPwd());
        psmt.setString(3, registerUser.getUserEmail());
        psmt.setInt(4, registerUser.getUserPin());

        result = psmt.executeUpdate();

    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        close(psmt);
    }

    return result == 0 ? false : true;
}
```

## 04-1.레벨 표시



```

public UserLevelDTO userLevel(int loggedUserID) {

    /* 1. 유저 정보 가져오기 */
    UserLevelDTO user = userService.userLevel(loggedUserID);

    /* 2. 기준에 맞게 레벨 update */
    //////////////////////////////////////
    // 레벨 1 습관수 0 성공횟수 0//
    // 레벨 2 습관수 2 성공횟수 4//
    // 레벨 3 습관수 3 성공횟수 5//
    // 레벨 4 습관수 4 성공횟수 6////////
    // 레벨 5 습관수 5이상 성공횟수 7이상//
    //////////////////////////////////////
    int userLevel = user.getUserLevel();
    int totalHabits = user.getNumOfHabits();
    int successOfChecks = user.getSuccessOfCheck();
    int successOfTimers = user.getSuccessOfTimer();

    int level = 1;

    if((successOfChecks + successOfTimers >= 7 ) && totalHabits >= 5 ) {
        level = 5;
    } else if((successOfChecks + successOfTimers >= 6 ) && totalHabits >= 4) {
        level = 4;
    } else if((successOfChecks + successOfTimers >= 5 ) && totalHabits >= 3) {
        level = 3;
    } else if((successOfChecks + successOfTimers >= 4 ) && totalHabits >= 2) {
        level = 2;
    } else {
        level = 1;
    }

    /* 3. 유저레벨에 변경이 생길 시, 유저 정보 업데이트
     * 3-1 View로 리턴값 변경
     * 3-2 DB update */
    if( userLevel != level ) {
        user.setUserLevel(level);
        userService.userLvUpdate(user);
    }

    return user;
}

```



## 04-1. 레벨 표시 업데이트


Lvl : 1



레벨: 1

레벨 정보

Username :

testl

Email :

test@test.com

Password :

a

PIN :

\*\*\*


수정


탈퇴

메인페이지

습관등록

습관삭제

마이페이지

2021년 04월 15일 목요일

# JUST HAB' IT

```

edLabel.addMouseListener(new MouseAdapter() {
    @Override
    public void mouseClicked(MouseEvent e){
        myUser_name.setEditable(true);
        myUser_name.setBackground(Color.LIGHT_GRAY);
        myUser_email.setEditable(true);
        myUser_email.setBackground(Color.LIGHT_GRAY);
        myUser_pwd.setEditable(true);
        myUser_pwd.setBackground(Color.LIGHT_GRAY);

        if(myUser_pin.getText().equals(String.valueOf(myUser.getUserPin()))){

            myUser.setUserName(myUser_name.getText());
            myUser.setUserEmail(myUser_email.getText());
            myUser.setUserPwd(myUser_pwd.getText());

            String result = userController.updateUser(myUser, myUser_name.getText());


            switch(result) {
                case "성공" :
                    JOptionPane.showMessageDialog(myPage, "회원정보 수정되었습니다!");
                    break;
                default :
                    JOptionPane.showMessageDialog(myPage, result);
            }

        }else {
            JOptionPane.showMessageDialog(myPage, "가입시 입력한 PIN을 입력해주세요");
            myUser_pin.setText("");
        }

        instruction.setText("<html>회원정보 <font color=red>수정</font>를 위한 <font color=blue>PIN</font>을 입력해주세요 :)</html>");
        instruction.setVisible(true);

        myUser_pin.setEditable(true);
        myUser_pin.setText("");
        myUser_pin.setBackground(Color.GRAY);
    }
});
    
```

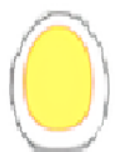
## 04-1. 레벨 표시 업데이트



Lvl : 1

2021년 04월 15일 목요일

# JUST HAB' IT



레벨 : 1

레벨 정보


Username : test1

Email : test@test.com


Password : a

회원정보 수정을 위한 PIN을 입력해주세요 :)

PIN :



수정



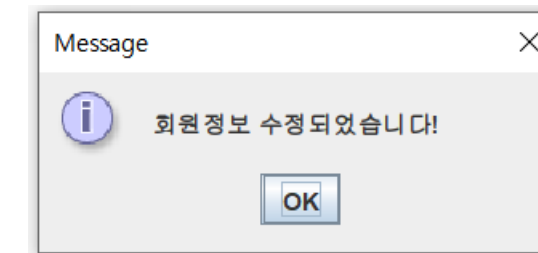
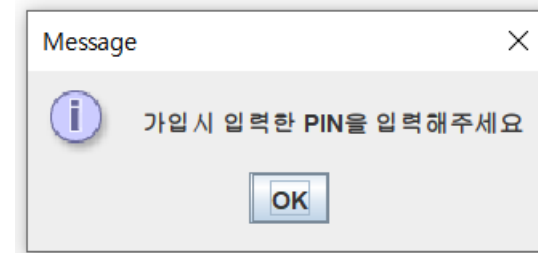
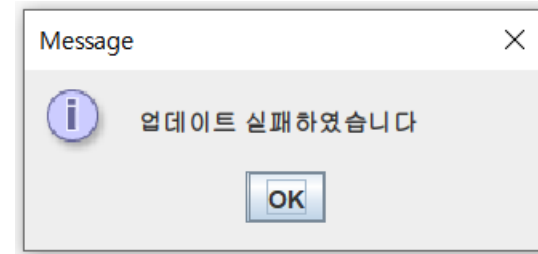
탈퇴

메인페이지

습관등록

습관악제

마이페이지



## 04-1. 레벨 표시 업데이트



```
UserLevelDTO myUserLevel = new TopPanel().userInfo(FirstFrame.loggedUserID);

for(int i = 1; i <= 5; i++) {

    Image level = new ImageIcon("image/" + i + "렙계란.PNG").getImage().getScaledInstance(65+(i*13), 70+(i*11), 0);

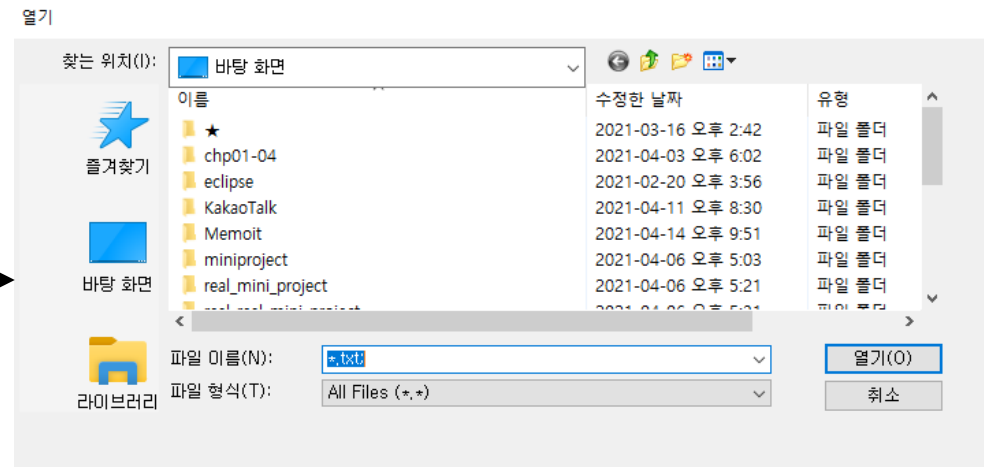
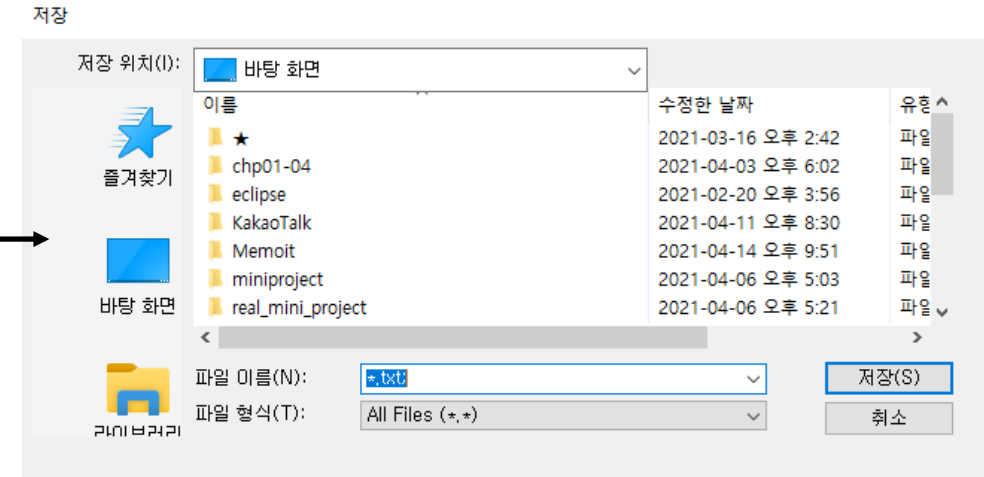
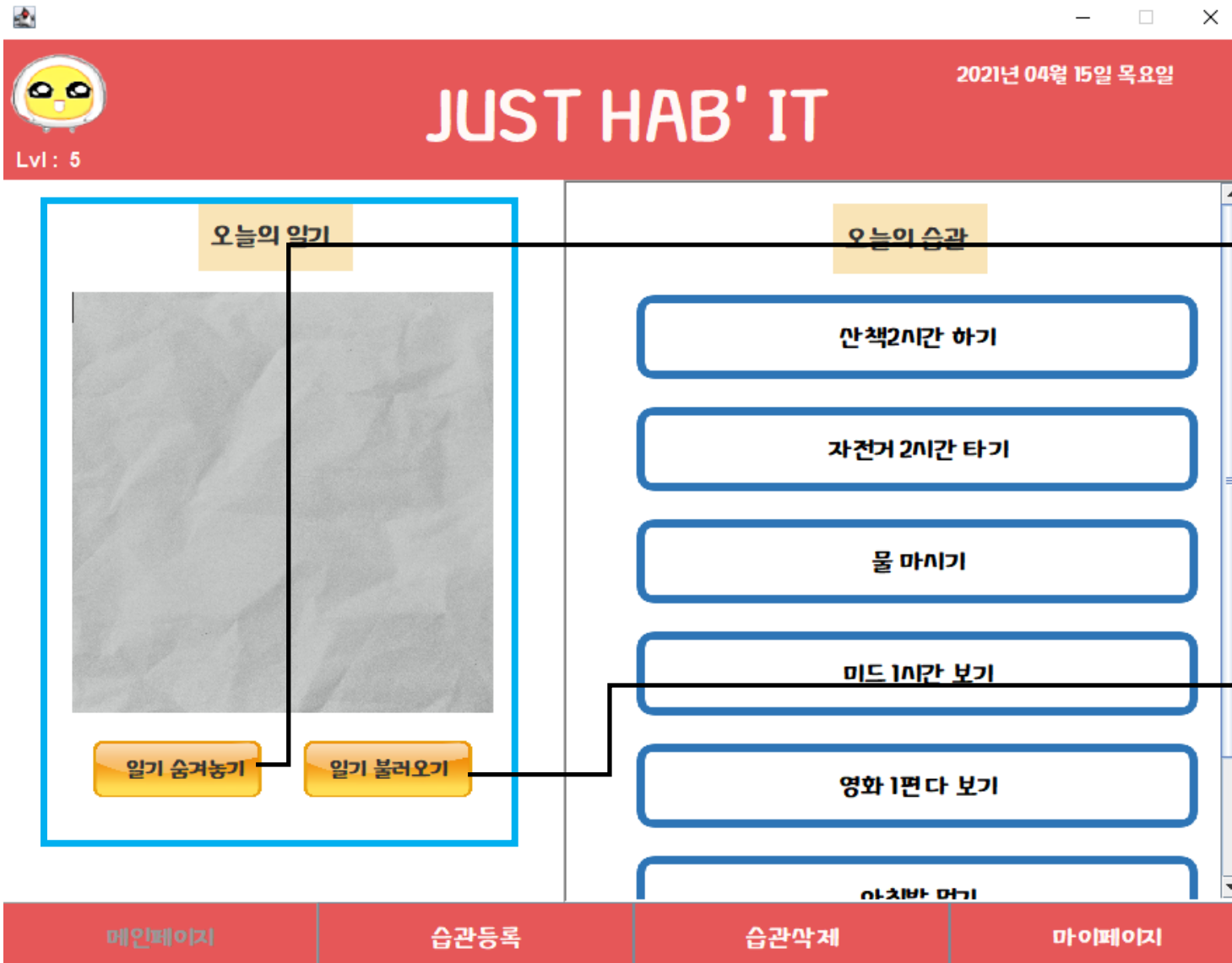
    JLabel levelImg = new JLabel(new ImageIcon(level));
    levelImg.setBounds((30 * i + (i-1) * 80), 40, 100, 100);
    this.add(levelImg);

    JLabel levelName;
    if (i == 1) {
        levelName = new JLabel("<html>" + i + " Level<br><br>"
                                + "<i>회원가입</i></html>");
        levelName.setBounds((50 * i), 165, 100, 100);
    } else {
        levelName = new JLabel("<html>" + i + " Level<br><br>"
                                + "습관 수 " + i + "<br>"
                                + "성공 수 " + (i+2)
                                + "</html>");
        levelName.setBounds((50 * i + (i-1) * 55), 70, 300, 300);
    }

    if(i == myUserLevel.getUserLevel()) {
        levelName.setForeground(Color.BLUE);
    }

    levelName.setFont(new Font("아디담돌", Font.BOLD, 8 + i*2));
    this.add(levelName);
}
```

## 04-2. 오늘의 일기





## 04-2. 오늘의 일기(파일 저장)

```
 JButton save = new JButton("일기 숨겨놓기");
 JButton open = new JButton("일기 불러오기");

 save.addActionListener(new ActionListener() {
     @Override
     public void actionPerformed(ActionEvent e){

         FileDialog dialog = new FileDialog(mf, "저장", FileDialog.SAVE);
         dialog.setFile("*.txt");
         dialog.setVisible(true);
         String path = dialog.getDirectory() + dialog.getFile();

         try {
             FileWriter writer = new FileWriter( path );
             writer.write(chat.getText());
             writer.close();
         } catch (Exception e2) {
             System.out.println("저장오류"+e2);
         }

         chat.setText("");

     }

 });
```

## 04-2. 오늘의 일기(파일 읽기)

```
open.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e) {  
        FileDialog dialog = new FileDialog(mf, "일기", FileDialog.LOAD);  
        dialog.setFile("*.txt");  
        dialog.setVisible(true);  
  
        String path = dialog.getDirectory() + dialog.getFile();  
        String diary = "";  
  
        if( dialog.getFile() == null ) return;  
        try {  
            FileReader reader = new FileReader( path );  
            int k;  
            for( ; ; ) {  
                k = reader.read();  
                if( k == -1 ) break;  
                diary += (char)k;  
            }  
            reader.close();  
        } catch (Exception e2) {  
            System.out.println("오류"+e);  
        }  
        chat.setText(diary);  
    }  
});
```

## 04-2. 습관 조회

```
<entry key="selectAllHabit">

    SELECT
        habit_id
      , habit_name
      , habit_type
      , habit_days
      , habit_goal
    FROM admin.hbt_register
   WHERE user_id = ?
      AND EXISTS (SELECT TO_CHAR(SYSDATE, 'DY')
                  FROM DUAL
                 WHERE habit_days LIKE '%' || TO_CHAR(SYSDATE, 'DY') || '%')

</entry>
```

## 04-2. 습관 조회

```
for(h = 0; h < allhabitList.size(); h++) {
    JButton box = new JButton();
    box.setSize(400, 60);
    box.setLocation(50, 80 * (h + 1));

    box.setOpaque(false);
    box.setContentAreaFilled(false);
    box.setBorderPainted(false);

    right.add(button1);

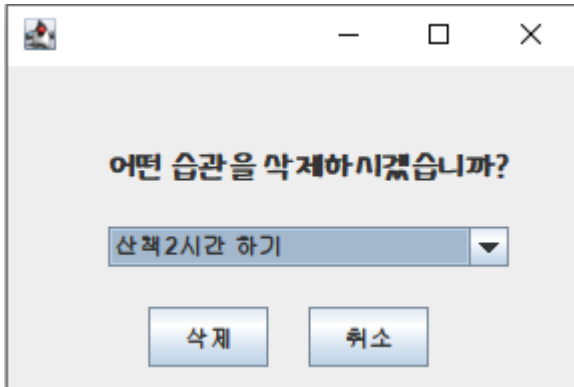
    right.add(box);

    int i = h;
    box.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent e) {

            userhabitid = allhabitList.get(i).getHabitID();
            System.out.println("-----"+userhabitid);
            if(allhabitList.get(i).getHabitType().equals("c")) {
                PanelChangeControl.changeFrame(mf, new CheckRecordView());
            } else {
                PanelChangeControl.changeFrame(mf, new TimeRecordView());
            }
        }
    });
    right.add(box);
}
```

## 04-2. 습관 삭제



```
 JButton delete = new JButton("삭제");
 JButton cancel = new JButton("취소");

 delete.setBounds(70, 120, 60, 30);
 cancel.setBounds(150, 120, 60, 30);

 this.add(delete);
 this.add(cancel);

 delete.addActionListener(new ActionListener() {
     @Override
     public void actionPerformed(ActionEvent e) {

         System.out.println(selectedHabbit);
         System.out.println(selectedHabitNum);
         deleteHabitID = allhabitList.get(selectedHabitNum).getHabitID();

         System.out.println(deleteHabitID);

         habitDayController.deleteHabitBy(deleteHabitID);

         mv.setVisible(false);
         PanelChangeControl.changeFrame(dh, new MainPage());

     }
 });

 cancel.addActionListener(e -> {
     dh.dispose();
 });
```

## 04-2. 습관 삭제

```

JLabel ask = new JLabel("어떤 습관을 삭제하시겠습니까?");
ask.setFont(new Font("THE외계인설명서", Font.BOLD, 15));
ask.setBounds(50, 40, 200, 20);
this.add(ask);

String[] habit = new String[allhabitList.size()];

for(int h = 0; h < allhabitList.size(); h++) {
    habit[h] = allhabitList.get(h).getHabitName();
}

JComboBox habitCombo = new JComboBox(habit);

habitCombo.setBounds(50, 80, 200, 20);
this.add(habitCombo);

ActionListener actionListener = new ActionListener() {
    public void actionPerformed(ActionEvent actionEvent) {
        System.out.println("Selected: " + habitCombo.getSelectedItem());
        System.out.println(". Index: " + habitCombo.getSelectedIndex());
        selectedHabbit = (String)habitCombo.getSelectedItem();
        selectedHabitNum = habitCombo.getSelectedIndex();
    }
};
habitCombo.addActionListener(actionListener);

```

```

JButton delete = new JButton("삭제");
JButton cancel = new JButton("취소");

delete.setBounds(70, 120, 60, 30);
cancel.setBounds(150, 120, 60, 30);

this.add(delete);
this.add(cancel);

delete.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {

        System.out.println(selectedHabbit);
        System.out.println(selectedHabitNum);
        deleteHabitID = allhabitList.get(selectedHabitNum).getHabitID();

        System.out.println(deleteHabitID);

        habitDayController.deleteHabitBy(deleteHabitID);

        mv.setVisible(false);
        PanelChangeControl.changeFrame(dh, new MainPage());

    }
});

cancel.addActionListener(e -> {
    dh.dispose();
});

```



## 04-3. 습관 추가

A-1) 무슨 습관을 들이고 싶나요?

아침에 일어나서 스트레칭 하기

메인으로
다음

A-2) 어떤 요일에 하고 싶나요?

☐ 일
 ☒ 월
 ☒ 화
 ☒ 수
 ☒ 목
 ☒ 금
 ☐ 토

이전
다음

```
public static HabitAddDTO habitAddDTO = new HabitAddDTO();
```

```
HabitAdd.habitAddDTO.setUserID(FirstFrame.LoggedUserID);
```

```
jbutton2.addMouseListener(new MouseAdapter() {
    @Override
    public void mouseClicked(MouseEvent e) {
        System.out.println(text.getText());

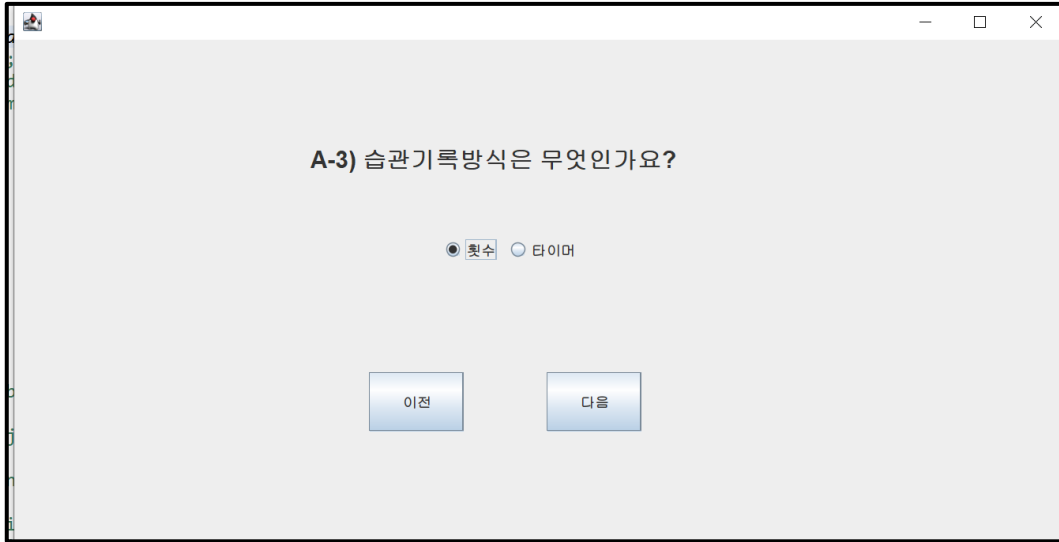
        HabitAdd.habitAddDTO.setHabitName(text.getText());
        PanelChangeControl.changeFrame(habitAdd , new HabitAdd3());
    }
});
```

```
jbutton2.addMouseListener(new MouseAdapter() {
    @Override
    public void mouseClicked(MouseEvent e) {

        String selectDay = "";

        for (int i = 0; i < buttons.length; i++) {
            if(buttons[i].isSelected()) {
                selectDay += buttons[i].getText();
            }
        }
        System.out.println("strDay : " + selectDay);
        HabitAdd.habitAddDTO.setHabitDays(selectDay);
    }
});
```

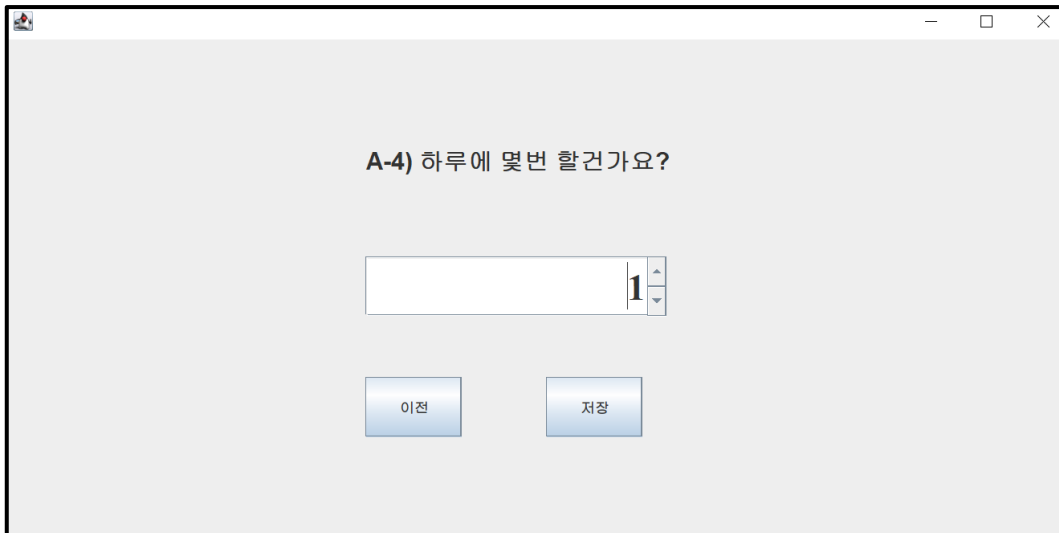
## 04-3. 습관 추가



A-3) 습관기록방식은 무엇인가요?

☒ 횟수 ☐ 타이머

이전 다음



A-4) 하루에 몇번 할건가요?

1

이전 저장

```
public static HabbitAddDTO habbitAddDTO = new HabbitAddDTO();
    ...
```

```
HabbitAdd.habbitAddDTO.setUserID(FirstFrame.loggedUserID);
```

```
JRadioButton count = new JRadioButton("횟수");
count.setActionCommand("c");
```

```
JRadioButton timer = new JRadioButton("타이머");
timer.setActionCommand("t");
```

```
if(HabbitAdd.habbitAddDTO.getHabitType() == "c") {
    label1.setText("A-4) 하루에 몇번 할건가요?");
    habbitAdd5.add(label1);
} else {
    label1.setText("A-4) 하루에 몇시간 할건가요?");
    habbitAdd5.add(label1);
}
```



# 04-3. 습관 추가

habit_id	habit_name	habit_type	habit_days	habit_goal
7	산책2시간 하기	t	수목금	2
3	자전거 2시간 타기	t	월화수목	2
8	물 마시기	c	수목금	5
9	미드 1시간 보기	t	수목금	1
6	영화 1편 다 보기	c	월화수목	1
10	아침밥 먹기	c	수목금	1
31	아침에 일어나서 스트레칭 하기	c	월화수목금	1

USER_ID	HABIT_ID	HABIT_NAME	HABIT_TYPE	HABIT_DAYS	HABIT_GOAL
10	10	17 운동하기	c	일월화	5
11	15	27 조깅 1시간 하기	t	수목금	1
12	1	32 아침등산	c	일	1
13	15	24 미드 1시간 보기	c	수목금	3
14	15	30 스트레칭 2번 하기	c	수목금	2
15	15	28 아침밥 먹기	c	수목금	1
16	10	34 테스트입니다	c	목	2
17	10	35 테스트2번입니다	t	목	1
18	15	29 물 마시기	c	수목금	1
19	10	33 아침밥먹기	c	목	1
20	1	10 아침밥 먹기	c	수목금	1
21	15	25 조깅 1시간 하기	t	목	1
22	1	31 아침에 일어나서 스트레칭 하기	c	월화수목금	1

```
<entry key="insertAllHabbit">
  INSERT

    INTO admin.hbt_register (
      user_id
      , habit_id
      , habit_name
      , habit_type
      , habit_days
      , habit_goal
    )

    VALUES (
      ?
      , admin.seq_habit_id.nextval
      , ?
      , ?
      , ?
      , ?
    )
```

04-4.

# 습관기록 조회(습관정보, 이번 달 기록)



```
//등록된 습관정보 불러오기
registInfo.setHabitID(MainPage.userhabitid);
registInfo = habitInfoController.selectHabitInfo(registInfo);
```

```
//기존 등록된 습관이 있으면 횟수 출력하기
SimpleDateFormat todayRecord = new SimpleDateFormat("yy/MM/dd");
String existingRecordDaty = todayRecord.format(todayDate);
if(recordAndGoalList.get(existingRecordDaty)!=null) {
    habitCount.setText("    목표 : "+ registInfo.getHabitGoal() + "회 / 현재 : "
        +recordAndGoalList.get(existingRecordDaty).getCheck() + "회    ");
    checkCount = recordAndGoalList.get(existingRecordDaty).getCheck();
}
```

```
//출력할 이번달 종합기록 검색
totalRecord.setHabitId(registInfo.getHabitID());
totalRecord.setTodayMonth(thisMonth);
totalRecord = habitInfoController.monthTotalController(totalRecord);
//습관 실시한 일수
totalDate = totalRecord.getDateCount();
//습관 총 횟수
totalCheck = (int)totalRecord.getRecordSum();
infoText.setText("\n \n \n
    + accomon + "일\n \n
    + totalDate + "일 \n \n
    + totalCheck + "회");
infoPanel.add(infoText);
```

이번 달 기록 \n \n \n  
 달성한 일수 : "  
 실시한 일수 : "  
 실시한 횟수 : "

## 04-4. 습관기록 조회(날짜별)



```
private Map<String,HabitRecordDTO> recordAndGoalList = null; //습관의 날짜별 기록 조회결과
recordAndGoalList = habitInfoController.selectRecordGoal(checkRecord);

//클릭시 그날 기록 출력하기
for(int i = 0; i < calArr.size(); i++) {
    searchDate = checkYearMonth+"/"+dayButton[i].getText();
    if(recordAndGoalList.get(searchDate)==null) {
        String dialogDate = searchDate;
        dayButton[i].addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                JOptionPane.showMessageDialog(mf, dialogDate + "\n\n" + 0 + " / " + registInfo.getHabitGoal());
            }
        });
    } else {
        String dialogDate = searchDate;
        dayButton[i].addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                JOptionPane.showMessageDialog(mf, dialogDate + "\n\n" + recordAndGoalList.get(dialogDate).getCheck()
                    + " / " + registInfo.getHabitGoal());
            }
        });
    }
}
}
```

## 04-4. 습관기록 조회(날짜별)

```
public Map<String,HabitRecordDTO> selectCheckRecordGoal(Connection con, HabitRecordDTO recordInfo) {
    PreparedStatement pstmt = null;
    ResultSet rset = null;
    HabitRecordDTO row = null;
    Map<String,HabitRecordDTO> userRecordGoalList = null;
    String query = prop.getProperty("selectCheckGoalRecord");

    try {
        pstmt = con.prepareStatement(query);
        pstmt.setInt(1, recordInfo.getHabitId());
        rset = pstmt.executeQuery();

        userRecordGoalList = new HashMap<>();

        while(rset.next()) {
            row = new HabitRecordDTO();
            String doDate = rset.getString("do_date");
            row.setCheck(rset.getInt("count_check"));
            row.setHabitGoal(rset.getInt("habit_goal"));
            userRecordGoalList.put(doDate, row);
        }
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        close(rset);
        close(pstmt);
    }

    return userRecordGoalList;
}
```

## 04-4. 습관기록 조회(날짜별)



```
private Map<String,HabitRecordDTO> recordAndGoalList = null; //습관의 날짜별 기록 조회결과
recordAndGoalList = habitInfoController.selectRecordGoal(checkRecord);

//클릭시 그날 기록 출력하기
for(int i = 0; i < calArr.size(); i++) {
    searchDate = checkYearMonth+"/"+dayButton[i].getText();
    if(recordAndGoalList.get(searchDate)==null) {
        String dialogDate = searchDate;
        dayButton[i].addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                JOptionPane.showMessageDialog(mf, dialogDate + "\n\n" + 0 + " / " + registInfo.getHabitGoal());
            }
        });
    } else {
        String dialogDate = searchDate;
        dayButton[i].addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                JOptionPane.showMessageDialog(mf, dialogDate + "\n\n" + recordAndGoalList.get(dialogDate).getCheck()
                    + " / " + registInfo.getHabitGoal());
            }
        });
    }
}
}
```

## 04-4. 습관기록 저장



```
saveLabel.addMouseListener(new MouseAdapter() {
    @Override
    public void mouseClicked(MouseEvent e) {
        SimpleDateFormat todayDateFormat = new SimpleDateFormat("yy/MM/dd");
        accomon = 0;

        if(checkCount == 0) {
            JOptionPane.showMessageDialog(mf, "등록할 기록 없음");
        } else {

            //습관기록을위한 기본정보(유저ID,습관ID,오늘날짜)

            checkRecord.setUserId(registInfo.getUserID()); //유저아이디
            checkRecord.setHabitId(registInfo.getHabitID()); // 습관아이디
            checkRecord.setCheck(checkCount); // 체크횟수
            today = todayDateFormat.format(todayDate);
            checkRecord.setDoDate(today); // 오늘날짜

            //날짜에 등록된 기록이 없으면 insert, 있으면 update
            int result = habitInfoController.dateSelectController(checkRecord);
            if(result==0){
                habitInfoController.insertCheckController(checkRecord);
                JOptionPane.showMessageDialog(mf, "신규 저장 성공");
            } else {
                habitInfoController.updateCheckController(checkRecord);
                JOptionPane.showMessageDialog(mf, "기록 갱신 성공");
            }
        }
    }
});
```

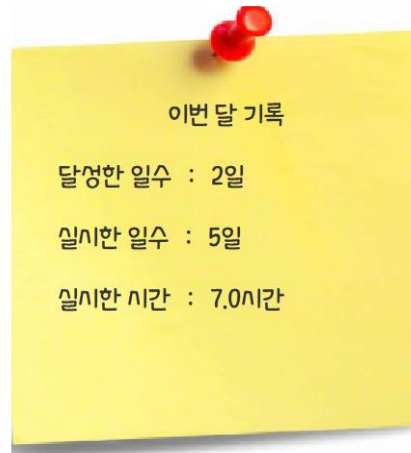


## 04-4. 습관기록 조회(타이머)



안책2시간 하기 목표: 2시간 / 00 : 00 : 00 : 00 [START] [PAUSE] [RESET] [저장]

04월					
01	02	03	04	05	06
07	08	09	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30



```
/**
 * @author
 * 타이머 시작, 정지, 종료
 */
class ButtonListener implements ActionListener{

    @Override
    public void actionPerformed(ActionEvent e) {
        String s = e.getActionCommand();

        if(s.equals("START")) {
            start.setEnabled(false);
            pause.setEnabled(true);
            reset.setEnabled(false);

            display = new Thread(new Runnable() {

                @Override
                public void run() {

                    hr = Integer.parseInt(hbtHr.getText());
                    mm = Integer.parseInt(hbtMin.getText());
                    ss = Integer.parseInt(hbtSec.getText());
                    ms = Integer.parseInt(hbtMillsec.getText());

                    while(display == Thread.currentThread()) {

                        hr = t / (60*60*100) % 24;
                        mm = t / (60*100) % 60;
                        ss = t / 100 % 60;
                        ms = t % 100;

                        try {
                            Thread.sleep(10);

                            hbtHr.setText(String.format("%02d", hr));
                            hbtMin.setText(String.format("%02d", mm));
                            hbtSec.setText(String.format("%02d", ss));
                            hbtMillsec.setText(String.format("%02d", ms));
                            t++;
                        } catch (InterruptedException e1) {
                        }
                    }
                }
            });
            display.start();
        }
        else if(s.equals("PAUSE")) {
            start.setEnabled(true);
            pause.setEnabled(false);
            reset.setEnabled(true);

            display = null;
        }
        else if(s.equals("RESET")) {
            start.setEnabled(true);
            pause.setEnabled(false);
            reset.setEnabled(false);

            hbtHr.setText("00");
            hbtMin.setText("00");
            hbtSec.setText("00");
            hbtMillsec.setText("00");
            t=0;
        }
    }
}
```



# Thank you

