

L

J

**PROJECT 1
COMPANY
MANAGEMENT
SYSTEM**

나현 / 박원희 / 장묘주 / 이기철

[] CONTENTS

/ PLAN

/ MVC

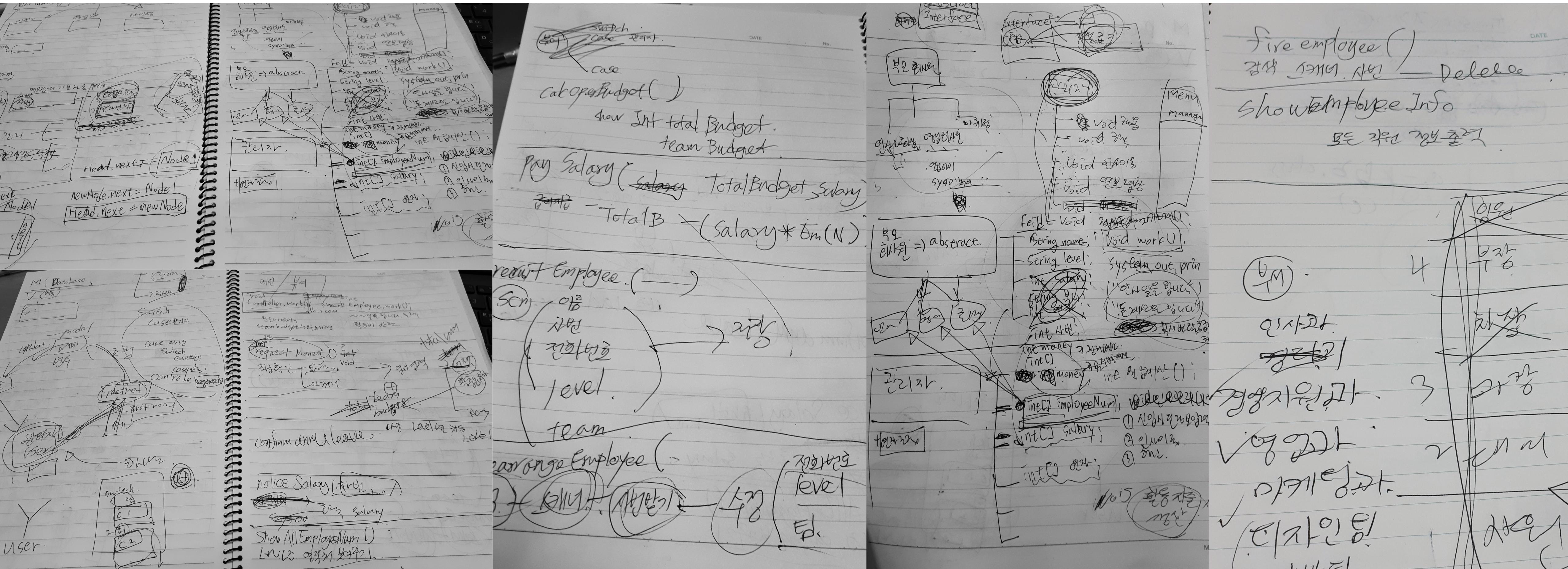
/ UML DIAGRAM

/ PROCESS

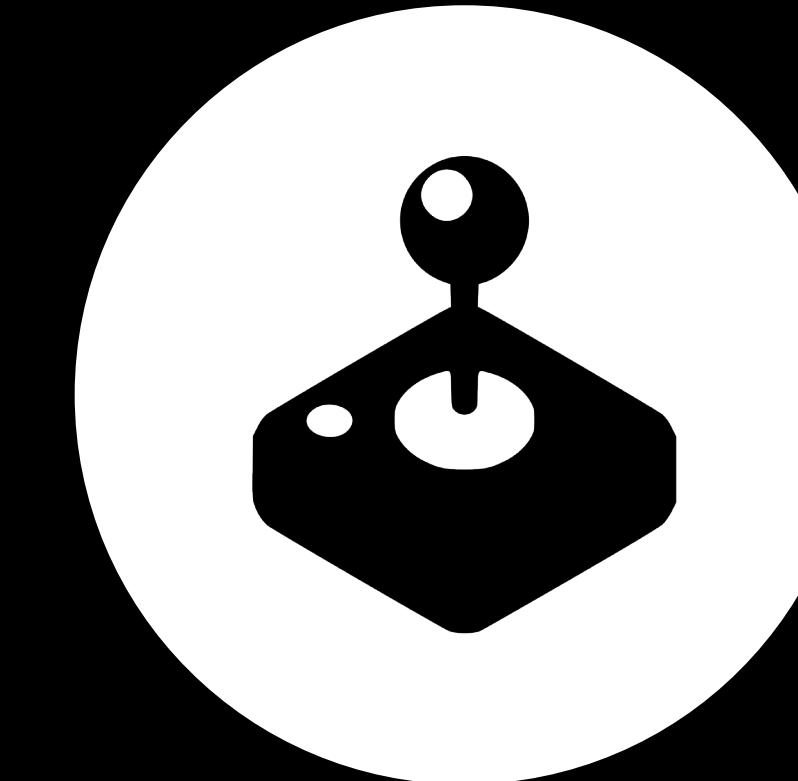
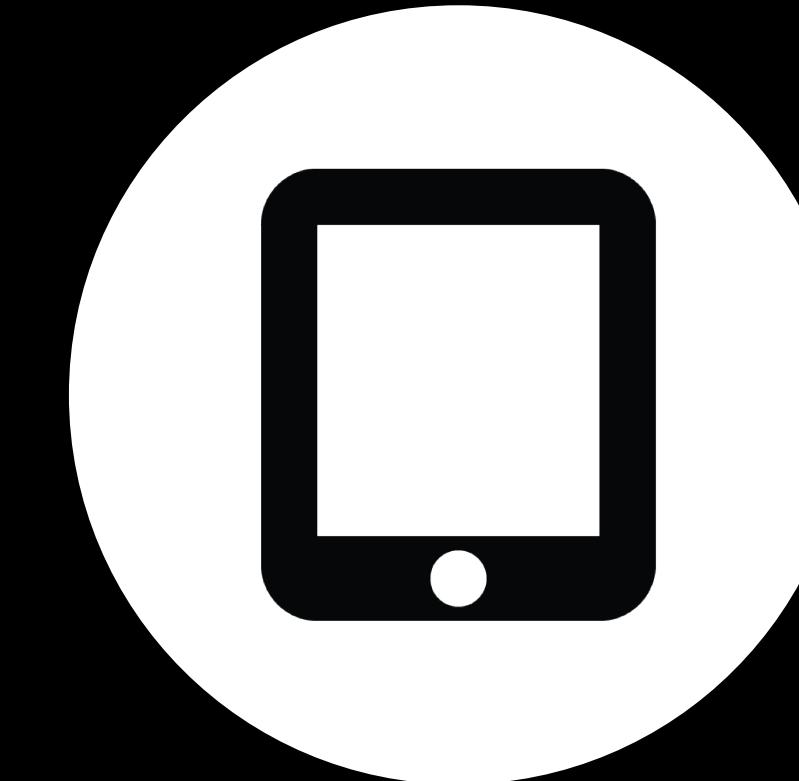
/ PRODUCTION

[FIRST PLAN OF SYSTEM]

프로젝트의 첫단계인 계획단계입니다. 어떠한 주제로 무엇을 수행하는 시스템을 개발할지 논의 한 후 시스템이 어떠한 기능과 서비스를 제공 해야하는지 기술하는 단계를 거쳤습니다. 일반적인 회사 매니지먼트 프로그램이 어떠한 기능을 가지는지 이해하고 담당할 작업의 리스트를 작성하여 구분하였습니다.



MVC



Model

- 정보
- / 직원정보
- / 인사정보
- / 회사정보
- / 재무정보

View

- 사용자가 보게 되는 화면
- / 관리자용 뷰
- / 사원용 뷰

Controller

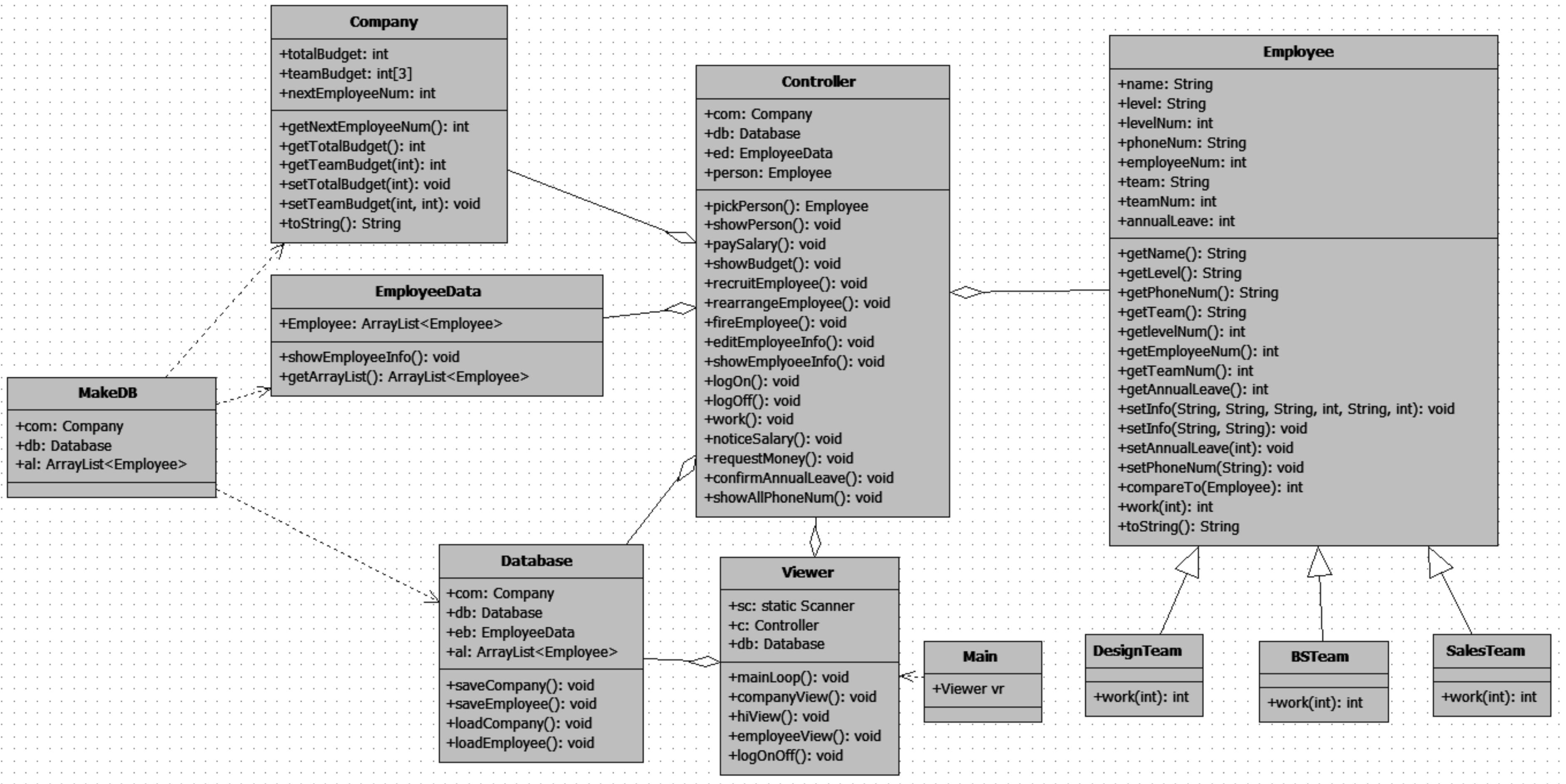
- 기능과 동작
- / 재무계산 기능
- / 근태관리 기능
- / 각종 행정제어

{

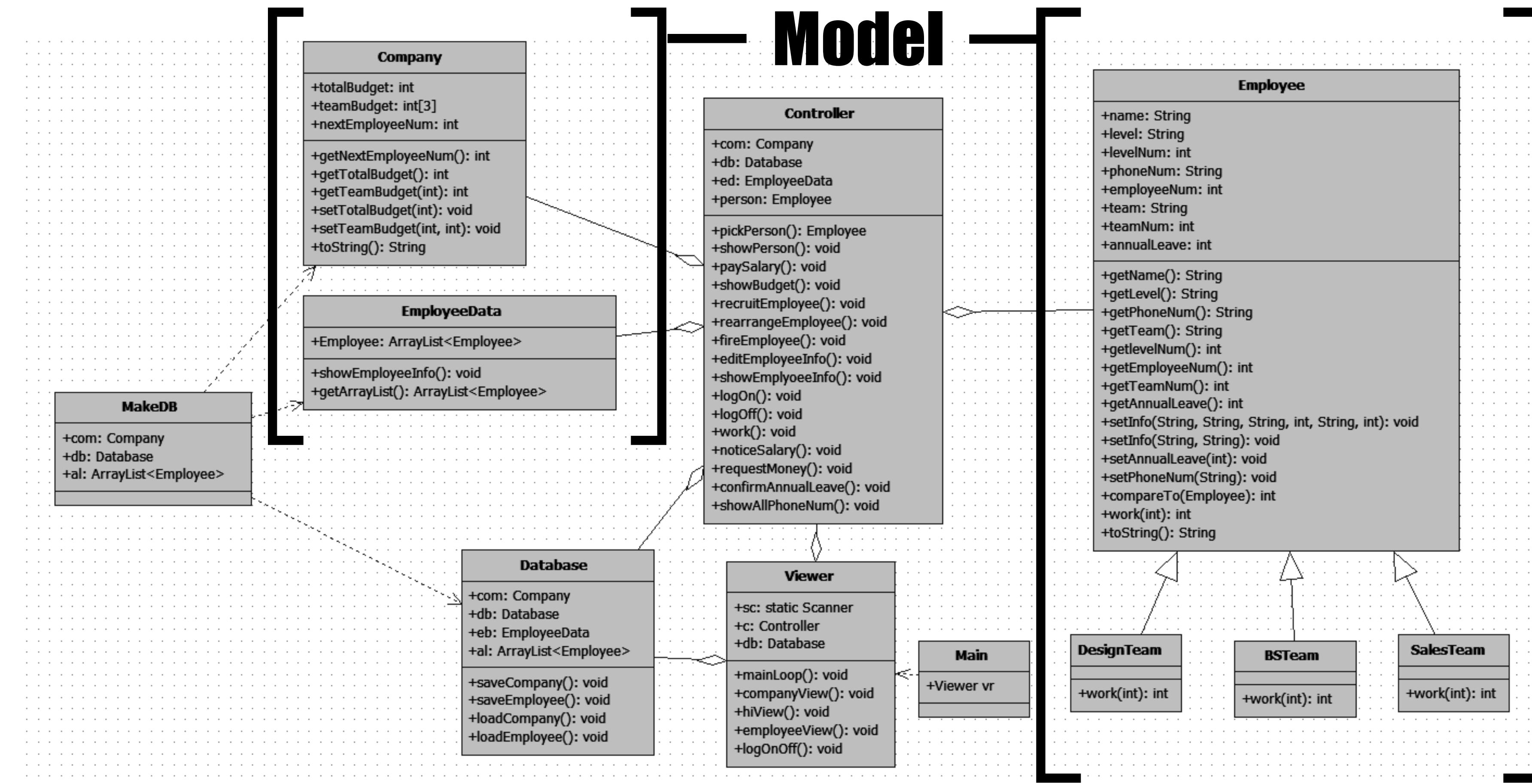
,

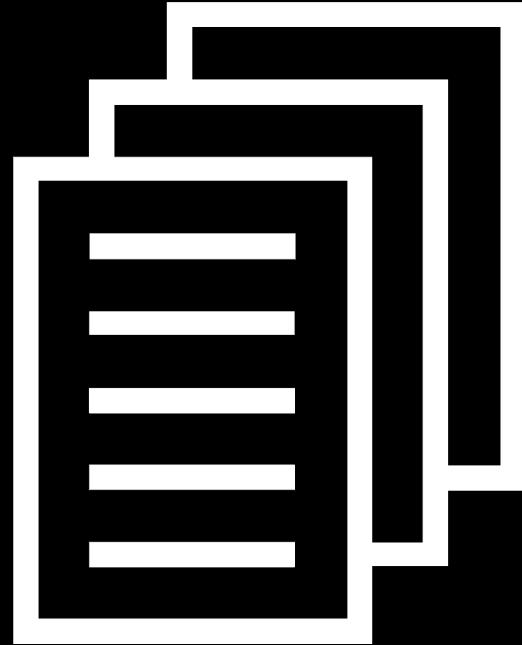
}

[UML DIAGRAM]



[UML DIAGRAM]





/ Model

```
public abstract class Employee
    implements Serializable,
    Comparable<Employee> {
    private String name;
    private String level;
    private int levelNum;
    private String phoneNum;
    private int employeeNum;
    private String team;
    private int teamNum;
    private int annualLeave;
    .
    .
}
```

사원의 어트리뷰트

```
public String toString() {
    return " 이름 : " + name +
        "\n 직급 : " + level +
        "\n 부서 : " + team +
        "\n 전화번호 : " + phoneNum +
        "\n 사번 : " + employeeNum ; }
```

사원의 **toString**

```
public Employee(String name, String level,
    String phoneNum, int employeeNum,
    String team, int annualLeave) {
    this.name = name;
    this.level = level;
    if (this.level.equals("부장"))
        this.levelNum = 4;
    else if (this.level.equals("과장"))
        this.levelNum = 3;
    else if (this.level.equals("대리"))
        this.levelNum = 2;
    else if (this.level.equals("사원"))
        this.levelNum = 1;
    this.phoneNum = phoneNum;
    this.employeeNum = employeeNum;
    this.team = team;
    if (team.equals("디자인팀"))
        teamNum = 2;
    else if (team.equals("경영지원팀"))
        teamNum = 1;
    else if (team.equals("영업팀"))
        teamNum = 0;
    this.annualLeave = annualLeave; }
```

생성자

```
public int compareTo(Employee e) {
    if
        (teamNum * 10 + levelNum > e.teamNum * 10 + e.levelNum)
            return -1;
    else if
        (teamNum * 10 + levelNum < e.teamNum * 10 + e.levelNum)
            return 1;
    return 0; }
```

ArrayList 정렬

오버라이딩을 이용한 재정의

```
public abstract int work(int teamBudget);

public int work(int teamBudget) {
    if(teamBudget < 50000) {
        System.out.println("예산이 부족합니다.");
        return 0;
    }
    else {
        System.out.println("영업 실적 목표 달성!");
        return -50000;
    }
}

public int work(int teamBudget) {
    if(teamBudget < 10000) {
        System.out.println("예산이 부족합니다.");
        return 0;
    }
    else {
        System.out.println("일해라 노예들아.");
        return -10000;
    }
}

public int work(int teamBudget) {
    if(teamBudget < 1000000) {
        System.out.println("예산이 부족합니다.");
        return 0;
    }
    else {
        System.out.println("열심히 디자인을 합니다.");
        return -100000;
    }
}
```

/ Model

컴퍼니의 어트리뷰트

```
public class Company implements Serializable{
    private int totalBudget;
    private int teamBudget[] = new int[3];
    private int nextEmployeeNum = 1;
}
```

컴퍼니의 toString

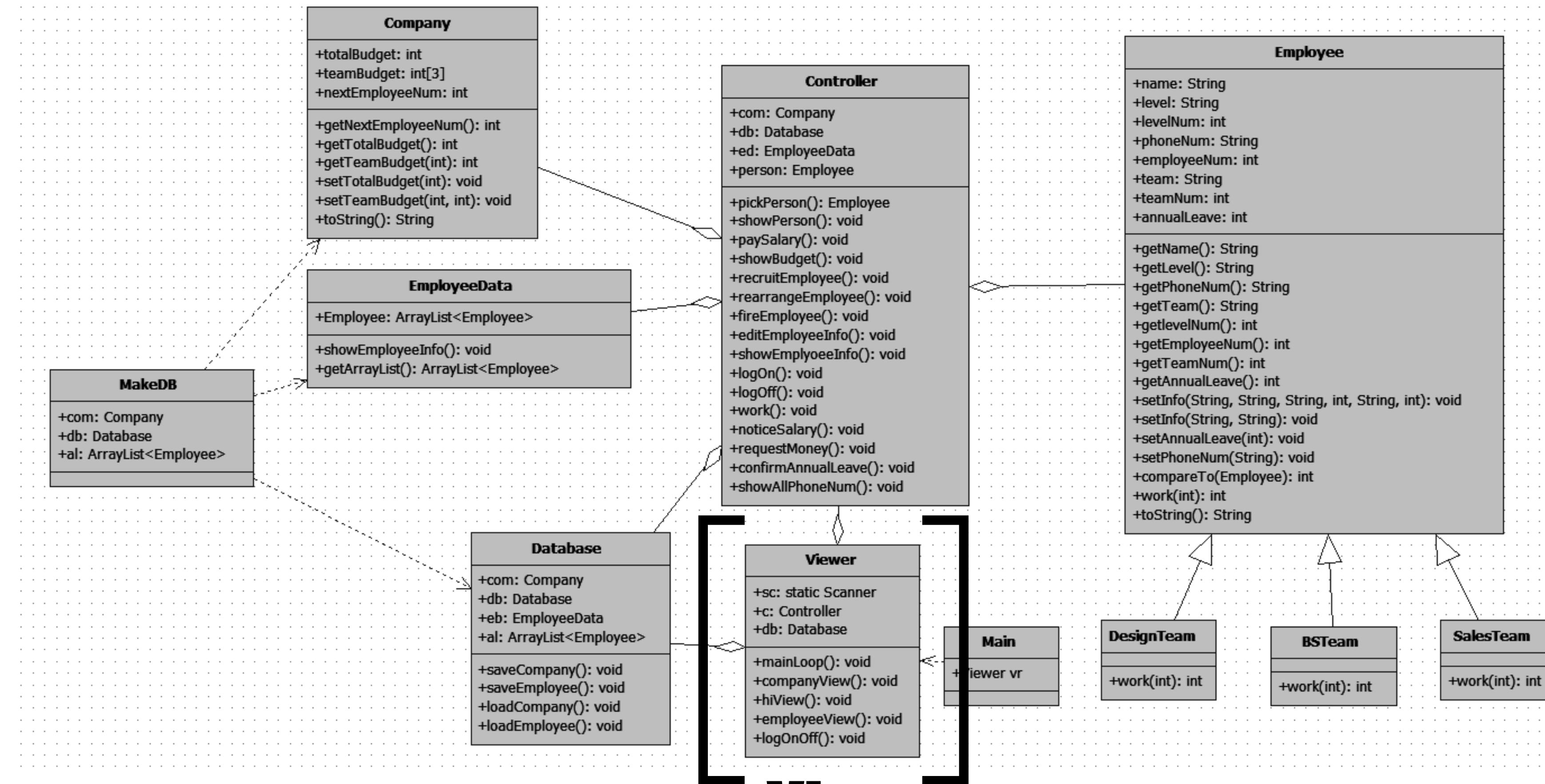
```
public String toString() {
    return "└ 기업 총예산 : " + totalBudget +
        "₩n └ 영업팀 예산 : " + teamBudget[0] +
        "₩n └ 경영지원팀 예산 : " + teamBudget[1] +
        "₩n └ 디자인팀 예산 : " + teamBudget[2]; }
```

ArrayList를 저장하기 위한 클래스

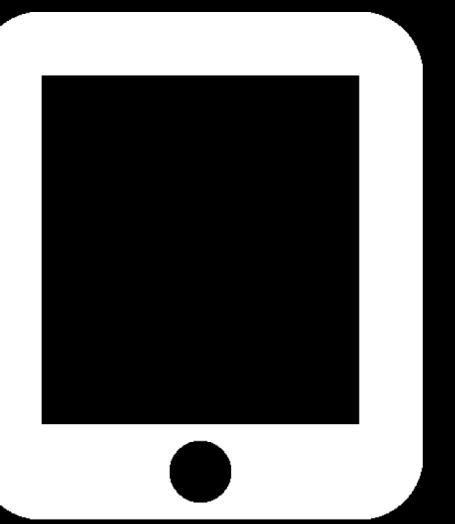
```
public class EmployeeData implements Serializable{
    ArrayList<Employee> a = new ArrayList<Employee>();
```

원희의 DB 설명

[UML DIAGRAM]



View

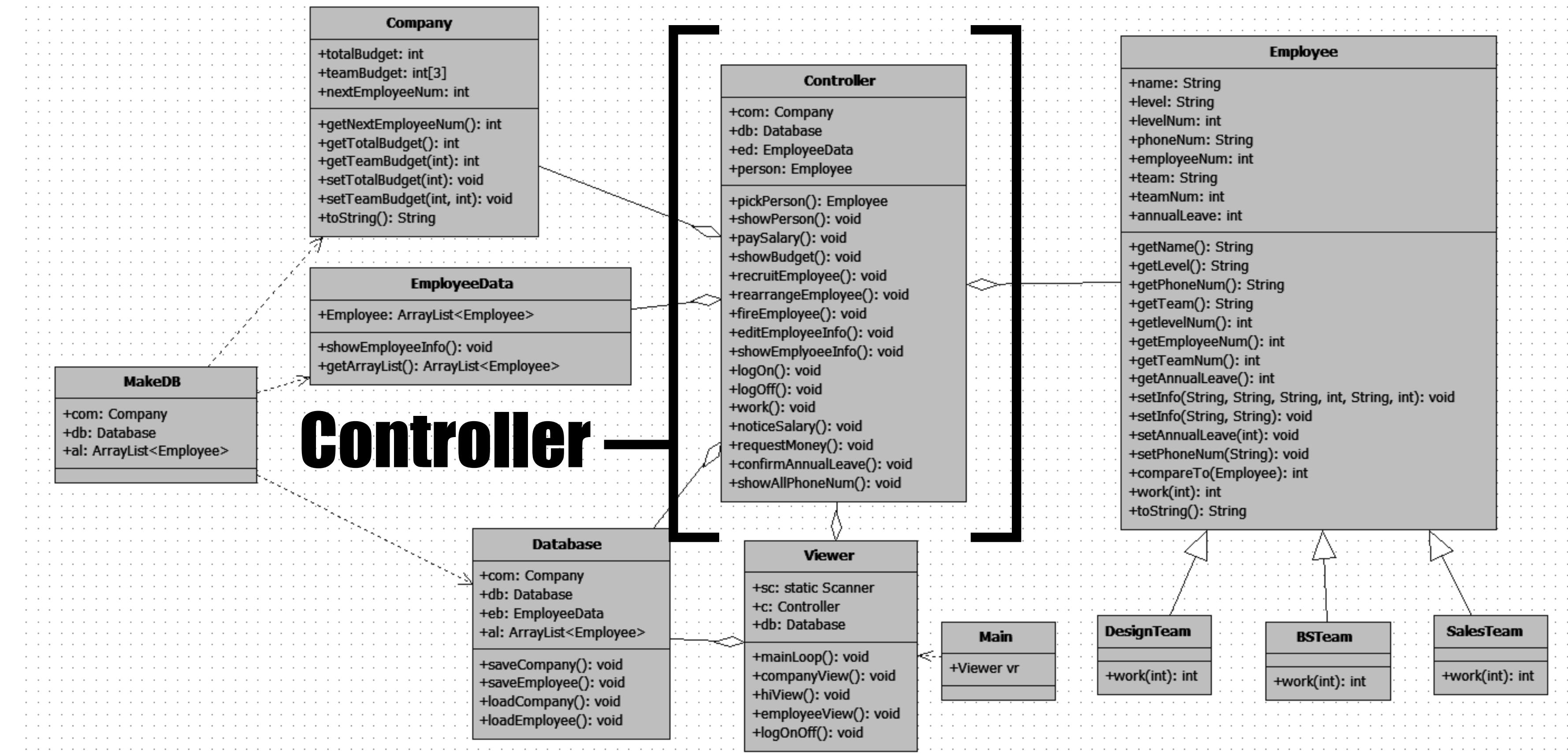


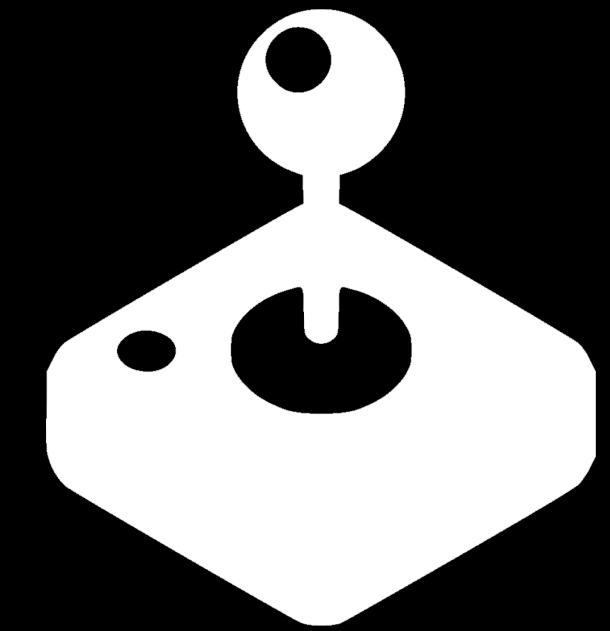
/ View

```
void companyView() {
    while (true) {
        System.out.println();
        System.out.println(" └ 관리자권한접속");
        System.out.println(" | ");
        System.out.println(" └ 메뉴를 선택해주세요.");
        System.out.println(" └ 1. 회사총액 및 활동비용");
        System.out.println(" └ 2. 월급지급시스템");
        System.out.println(" └ 3. 인사관리시스템");
        System.out.println(" └ 4. 모든사원정보확인");
        System.out.println(" └ 5. 이전메뉴로 이동");
        int sel = sc.nextInt();
        if (sel == 5)
            break;
        switch (sel) {----중략----}
    }
}

void employeeView() {
    if (c.pickPerson() == null)
        return;
    while (true) {
        System.out.println();
        System.out.println(" └ 사원권한접속");
        c.showPerson();
        System.out.println(" | ");
        System.out.println(" └ 메뉴를 선택해주세요.");
        System.out.println(" └ 1. 근태관리시스템");
        System.out.println(" └ 2. 업무보고");
        System.out.println(" └ 3. 업무비용요청");
        System.out.println(" └ 4. 연차신청시스템");
        System.out.println(" └ 5. 급여조회시스템");
        System.out.println(" └ 6. 비상연락망확인");
        System.out.println(" └ 7. 이전메뉴로 이동");
        int sel = sc.nextInt();
        if (sel == 7)
            break;
        switch (sel) {----중략----}
    }
}
```

[UML DIAGRAM]





/ Controller

public Employee pickPerson()

/ 사번 조회 기능

public void paySalary()
void showBudget()
void recruitEmployee()
void editEmployeeInfo()
void rearrangeEmployee()
void fireEmployee()
void showEmployeeInfo()

/ 급여지급자동화
/회사의 재무상태 확인
/신입사원 정보입력
/기존사원 정보수정
/인사이동
/사원해고
/모든 사원 정보보기

public void showPerson()
void logOn()
void logOff()
public void work()
void requestMoney()
void noticeSalary()
void confirmAnnualLeave()
void showAllPhoneNum()

/로그인한 사번을 토대로 시스템 인사
/근태확인_출근
/근태확인_퇴근
/업무보고 및 경비와 매출
/업무비용 요청
/급여확인
/연차확인 및 신청
/비상연락망확인

원희

사번 조회 기능

```
public Employee pickPerson() {
    System.out.print("→ 조회용 사번 입력: ");
    Iterator<Employee> iter = ed.getArrayList().iterator();
    int eNum = Viewer.sc.nextInt();
    while (iter.hasNext()) {
        person = iter.next();
        if (eNum == person.getEmployeeNum())
            return person;
    }
    person = null;
    System.out.println("→ 존재하지 않는 사번입니다.");
    return person;
}
```

나현

모든 사원정보 확인

```
void showEmployeeInfo() {
    Iterator<Employee> iter = ed.getArrayList().iterator();
    System.out.println();
    while (iter.hasNext())
        System.out.println(iter.next());
}
```

비상연락망확인

```
void showAllPhoneNum() {
    Iterator<Employee> iter = ed.getArrayList().iterator();
    Employee e;
    System.out.println();
    while (iter.hasNext()) {
        e = iter.next();
        System.out.println(e.getName() + " " + e.getLevel() + " "
            + e.getPhoneNum() + " " + e.getTeam());
    }
}
```

업무보고 및 매출+경비 계산

```
public void work() {
    System.out.println();
    int money = person.work(com.getTeamBudget
        (person.getTeamNum()));
    com.setTeamBudget(person.getTeamNum(), money);
    if(money != 10000)
        com.setTotalBudget(money * -2);
}
```

연차 조회 및 신청

```
void confirmAnnualLeave() {
    System.out.println();
    System.out.println("└ 연차신청시스템");
    System.out.println("└ 사용하실 연차 일수를 입력하세요.");
    int i = Viewer.sc.nextInt();
    if (person.getAnnualLeave() >= i) {
        System.out.println("→ 연차가 승인되었습니다.");
        person.setAnnualLeave(i * -1);
    } else
        System.out.println("→ 연차일수 부족으로 거절되었습니다.");
}
```

부서별 경비요청

```
void requestMoney() {  
    System.out.println("→ 업무비용요청 권한처리중입니다.");  
    if (person.getLevel().equals("부장")) {  
        System.out.println("└ 권한이 확인되었습니다.");  
        System.out.println("└ 필요한 금액을 입력하세요.");  
        int money = Viewer.sc.nextInt();  
        if (money > com.getTotalBudget()) {  
            System.out.println("└ " + com.getTotalBudget() + "원 이하로 입력해주세요.");  
        } else if (money <= com.getTotalBudget()) {  
            com.setTotalBudget(money * -1);  
            com.setTeamBudget(person.getTeamNum(), money);  
            System.out.println("└ " + money + "원 지급 완료되었습니다.");  
        } else {  
            System.out.println("└ 권한이 없습니다.");  
        }  
    }  
}
```

기찰

묘주

급여지불

```
public void paySalary() {  
    int salary = 2000000;  
    Iterator<Employee> iter = ed.getArrayList().iterator();  
    System.out.println();  
    System.out.println("└ 급여지불자동시스템");  
    Employee e;  
    while (iter.hasNext()) {  
        e = iter.next();  
        if (e.getlevelNum() == 4) {  
            System.out.println("└ " + e.getName() + " 부장에게 " +  
                (salary * 4) + "원을 지급합니다.");  
            com.setTotalBudget(salary * 4 * -1);  
        } else if (e.getlevelNum() == 3) {  
            System.out.println("└ " + e.getName() + " 차장에게 " +  
                (salary * 3) + "원을 지급합니다.");  
            com.setTotalBudget(salary * 3 * -1);  
        } else if (e.getlevelNum() == 2) {  
            System.out.println("└ " + e.getName() + " 대리에게 " +  
                (salary * 2) + "원을 지급합니다.");  
            com.setTotalBudget(salary * 2 * -1);  
        } else if (e.getlevelNum() == 1) {  
            System.out.println("└ " + e.getName() + " 사원에게 " +  
                (salary) + "원을 지급합니다.");  
            com.setTotalBudget(salary * 1 * -1);  
        }  
    }  
}
```

사원 해고

```
void fireEmployee() {  
    if(pickPerson() == null)  
        return;  
    System.out.println("└ " + person.getName() + "를 선택하셨습니다.");  
    ed.getArrayList().remove(person);  
    System.out.println("└ 해고를 통지하였습니다.");  
    System.out.println("└ 해당 사원의 데이터를 시스템에서 삭제 합니다.");  
}
```

신입사원 정보입력

```
void recruitEmployee() {
    String name, team, phoneNum, level;
    int emNum, annLeave;
    System.out.println();
    System.out.println(" └ 신입사원관리시스템");
    System.out.println(" | ");
    System.out.println(" └ 입사한 사원의 정보를 입력해주십시오.");
    System.out.print(" └ 이름: ");
    name = Viewer.sc.next();
    System.out.print(" └ 직급: ");
    level = Viewer.sc.next();
    emNum = com.getNextEmployeeNum();
    System.out.print(" └ 연락처: ");
    phoneNum = Viewer.sc.next();
    System.out.print(" └ 부서: ");
    team = Viewer.sc.next();
    System.out.print(" └ 연차정보: ");
    annLeave = Viewer.sc.nextInt();
    if (team.equals("디자인팀")) {
        ed.getArrayList().add(new DesignTeam(name, level, phoneNum, emNum, team, annLeave));
        Collections.sort(ed.getArrayList());
    } else if (team.equals("경영지원팀")) {
        ed.getArrayList().add(new BSTeam(name, level, phoneNum, emNum, team, annLeave));
        Collections.sort(ed.getArrayList());
    } else if (team.equals("영업팀")) {
        ed.getArrayList().add(new SalesTeam(name, level, phoneNum, emNum, team, annLeave));
        Collections.sort(ed.getArrayList()); } }
```

다음은

시연입니다!!