



---

# УПРАВЛЕНИЕ НА ПРОЕКТИ

---

База данни – тема No 21



MAY 31, 2023

ИЗГОТВИЛ: ПЕТЯ ЛИЧЕВА

Факултетен номер: ЗМІ0700022

## Съдържание

Обхват на модела. Дефиниране на задачата .....	2
Множества от същности и техните атрибути .....	2
Домейн на атрибутите .....	3
Връзки .....	3
Ключове .....	4
Правила и проверки .....	4
Е/R модел на данни .....	6
Релационен модел на данни .....	6
Таблицы .....	6
Извод .....	9
Схема на базата от данни .....	12
Изгледи .....	12
Employees view .....	12
Tasks view .....	13
Bugs view .....	13
Приложение за достъп до базата от данни .....	13
main() method .....	13
printMenu() method .....	13
execute() .....	14
Конзола с резултатните множества от заявките .....	15

## Обхват на модела. Дефиниране на задачите

Базата от данни за управление на проекти ще съхранява информацията за задачите, свързани с даден проект, както и за техните изпълнители.

Базата от данни ще съхранява информация за различни компании. Компанията ще се определя еднозначно от име на компанията, адрес, описание на дейността на компанията. Във всяка компания ще има различни екипи (минимум 1 и максимум 10).

Всеки екип ще се определя еднозначно от име на екипа, описание на дейността на екипа. Във всеки екип ще има минимум 1 служител (който ще е наследник на множеството от същности на потребителите) и всеки екип ще работи по различни проекти, а по всеки проект ще могат да работят много екипи.

Всеки потребител на системата се определя еднозначно от уникален номер, имейл, потребителско име (до 32 символа) и парола (до 32 символа). Потребителите ще са два вида – служители и любители.

Всеки служител се определя еднозначно от телефонен номер, адрес, заплата, екип, в който работят.

Всеки любител се определя еднозначно от описание на дейността, с която се занимават любителски.

Всеки проект се определя еднозначно от име, версия, описание, дата на пускане в употреба. Проектът ще се състои от много задачи като една задача ще може да участва в точно един проект.

Всяка задача се определя еднозначно от уникален номер, начална дата, статус (завършена или блокирана) и текстово описание. За някои задачи е дефиниран и краен срок (който не може да е преди датата на създаване на задачата). Задължително се съхранява информация за това кой потребител и на коя дата е дефинирал задачата. Бъговете са специални задачи, за които задължително се съхранява сценарий за възпроизвеждането им, както и номер на версията на приложението, в която са били забелязани за първи път. Задсачите биват – изпълними задачи или бъгове.

Всяка изпълнима задача се определя еднозначно от педполагаема крайна дата.

Всеки бърк се определя еднозначно от сценарий за възпроизвеждането им, номер на версията на приложението, в което са били забелязани за първи път.

Всяка блокирана задача се определя еднозначно от дата на блокиране, блокираща задача. Един потребител може да работи по няколко задачи, а по една задача могат да работят няколко потребителя едновременно. Дадена задача може да бъде блокирана от други задачи (тоест необходимо е те да бъдат завършени, за да може да се работи по нея) и една задача може да блокира няколко други задачи.

## Множества от същности и техните атрибути

Компании – име на компанията, адрес на компанията, описание на дейността на компанията

Екипи – име на компанията, име на екипа, описание на дейността на екипа  
Потребители – уникален номер, имейл, потребителско име и парола  
Служители – телефонен номер, адрес, заплата, екип, в който работят  
Любители – описание на дейността, с която се занимават любителски  
Проекти – име на проекта, версия на проекта, описание, дата на пускане в употреба  
Задачи – уникален номер на задачата, начална дата, статус, описание  
Изпълними задачи – уникален номер на задачата, предполагаема крайна дата  
Бъгове – уникален номер на задачата, сценарий за възпроизвеждането им, номер на версията на приложението, в което са били забелязани за първи път

## Домейн на атрибутите

Компании – име на компанията: низ, адрес: низ, описание на дейността на компанията: низ  
Екипи – име на компанията: низ, име на екипа: низ, описание на дейността на екипа: низ  
Потребители – уникален номер: цяло положително число, потребителско име: низ, имейл: низ и парола: низ  
Служители – уникален номер: цяло положително число, телефонен номер: низ, адрес: низ, заплата: число с плаваща запетая  
Любители – уникален номер: цяло положително число, описание на дейността, с която се занимават любителски: низ  
Проекти – име: низ, версия: низ, описание: низ, дата на пускане в употреба: дата  
Задачи – уникален номер: число, начална дата: дата, описание: низ  
Изпълними задачи – предполагаема крайна дата: дата  
Бъгове – сценарий за възпроизвеждането им: низ

## Връзки

**Потребители – Служители:** Множеството от потребители е корен на йерархия на наследственост, а множеството от служители е негов наследник (isa връзка)  
**Потребители – Любители:** Множеството от потребители е корен на йерархия на наследственост, а множеството от любители е негов наследник (isa връзка)  
**Потребители – Задачи:** Един потребител може да дефинира много задачи, но една задача може да е дефинирана от точно един потребител.  
**Компании – Екипи:** Една компания може да има между 1 и 10 екипа, а един екип може да участва в точно една компания (не се допуска екип без компания, за която да работи)  
**Екипи – Служители:** В един екип може да има много служители, но не може да няма нито един служител. Един служител може да работи в точно един екип.  
**Проекти – Задачи:** Един проект може да се състои от 1 или от много задачи, а една задача, може да е част само от един проект.  
**Екипи – Проекти:** Един екип може да работи по много проекти и по един проект могат да работят много екипи.  
**Любители – Задачи:** Един любител може да работи по много задачи и по една задача могат да работят много любители.

**Задачи – Изпълними задачи:** Множеството от задачи е корен на йерархия на наследственост, а множеството от изпълними задачи е негов наследник (isa връзка)

**Задачи – Бъгове:** Множеството от задачи е корен на йерархия на наследственост, а множеството от бъгове е негов наследник (isa връзка)

**Бъгове – Проекти:** Един бърг може да се забележи за пръв път в точно една версия на даден проект, но в една версия на проект може да се забележат много бъгове.

## Ключове

Компании – име на компанията: низ, адрес на компанията: низ уникално определят компанията.

Екипи – име на компанията: низ, име на екипа: низ уникално определят екипа.

Потребители – уникален номер: цяло число уникално определя потребителя.

Служители – уникален номер: цяло число уникално определя служителя.

Любители – уникален номер: цяло число уникално определя любителя.

Проекти – име на проекта: низ, версия на проекта: низ уникално определят проекта.

Задачи – уникален номер на задачата: цяло число уникално определя задачата.

Изпълними задачи – уникален номер на задачата: цяло число уникално определя изпълнимата задача.

Бъгове – уникален номер на задачата: цяло число уникално определя бърга.

## Правила и проверки

Компании – име на компанията (до 20 символа, не може да е NULL), адрес на компанията (до 100 символа, адресът да е записан в български стандарт за адреси – компаниите ще са само на територията на страната, не може да е NULL), описание на дейността на компанията (до 450 символа, може да е NULL)

Екипи – име на екипа (до 20 символа, не може да е NULL), описание на дейността на екипа (до 300 символа, може да е NULL)

Потребители – уникален номер (последователно, уникално, цяло число, не може да е NULL), потребителско име (до 32 символа – да съдържа букви и цифри и да започва с буква, не може да е NULL), имейл (валиден имейл адрес до 100 символа, не може да е NULL) и парола (до 32 символа – да съдържа букви, цифри и специални символи като \_ - . ! # @, не може да е NULL)

Служители – телефонен номер (валиден телефонен номер до 20 символа, не може да е NULL), адрес (до 100 символа, адресът да е записан в български стандарт адреси – служителите ще са само на територията на страната, може да е NULL), заплата (положително число с плаваща запетая, може да е NULL)

Любители – описание на дейността, с която се занимават любителски (до 100 символа, може да е NULL)

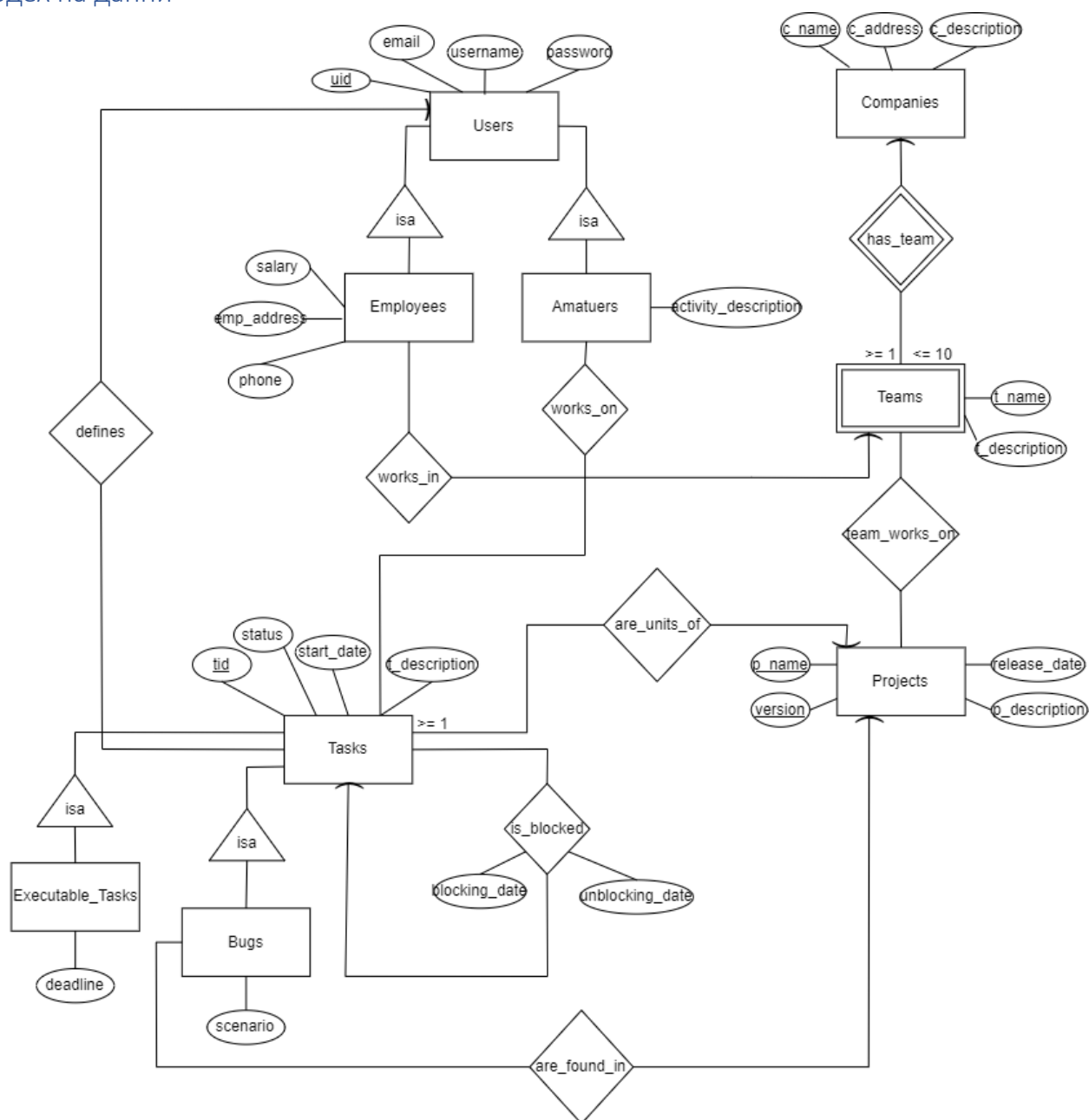
Проекти – име на проекта (до 30 символа, не може да е NULL), версия на проекта (до 20 символа, не може да е NULL), описание (до 300 символа, може да е NULL), дата на пускане в употреба (датата на пускане на проекта да не е преди 01.01.2023 година, не може да е NULL)

Задачи – уникален номер на задачата (поредно, уникално, цяло число, не може да е NULL), начална дата (датата да не е преди 01.01.2023 година, може да е NULL), статус (завършена или блокирана, може да е NULL в случаите, когато все още се изпълнява), описание (до 300 символа, може да е NULL), статус (завършена/незавършена, може да е NULL)

Изпълними задачи – предполагаема крайна дата (предполагаемата крайна дата да е поне месец след началната дата на задачата, може да е NULL)

Бъгове – сценарий за възпроизвеждането им (до 600 символа, не може да е NULL), номер на версията на приложението, в което са били забелязани за първи път (до 20 символа, не може да е NULL).

## E/R модел на данни



## Релационен модел на данни

### Таблицы

ISA йерархия Потребители

Релационен подход:

Users (uid: int (> 0), email: string (320), username: string (10), password: string (64))  
Employees (uid: int (> 0), salary: double (> 0.0), emp\_address: string (100), phone: string (>= 10 && <= 13)) Amateurs (uid: int (> 0), activity\_description: string (350))

#### ООП подход:

**Users** (uid: int (> 0), email: string (320), username: string (10), password: string (64))

**ПК:** uid (unique, not null)

Employees (uid: int (> 0), salary: double (> 0.0), emp\_address: string (100), phone: string (>= 10 && <= 13)) – таблицата отпада

Amateurs (uid: int (> 0), activity\_description: string (100)) – таблицата отпада

**Users\_Employees** (uid: int (> 0), email: string (320), username: string (10), password: string (64), salary: double (> 0.0), emp\_address: string (100), phone: string (>= 10 && <= 13), team\_id: int (> 0))

**ПК:** uid (unique, not null)

**FK:** Users (team\_id) -> Teams (tid)

**Users\_Amateurs** (uid: int (> 0), email: string (320), username: string (10), password: string (64), activity\_description: string (350))

**ПК:** uid (unique, not null)

#### NULL подход:

Users (uid: int (> 0), email: string (320), username: string (10), password: string (64), team\_id: int (> 0), salary: double (> 0.0), emp\_address: string (100), phone: string (>= 10 && <= 13), activity\_description: string (350))

Избирам ООП подхода за йерархията на потребителите със съкращаване на две от таблиците (Employees и Amateurs), защото те са нефункционални и не носят достатъчно информация за въпросните роли на потребителя. Не избирам релационния подход или NULL подхода, защото при бъдещи разширения на системата биха били лош вариант поради излишество (повторения) на данни. С оглед на възможните транзакции към базата този подход би бил най-удачен, защото в повечето случаи ще се обръщаме към само една от таблиците в модела. За потребителите ще генерираме 3 таблици.

#### ISA йерархия Задачи

##### Релационен подход:

**Tasks** (tid: int (> 0), status: string (10 – unfinished), start\_date: date (>= 01.01.2023), tdescription: string (350), uid: int (> 0), project\_id: int (> 0), btid: int (> 0))

**ПК:** tid (unique, not null)

**FK:** Tasks (uid) -> Tasks (uid)

**FK:** Tasks (project\_id) -> Projects (pid)

**FK:** Tasks (btid) -> Tasks (tid)

**Executable\_Tasks** (tid: int (> 0), deadline: date (>= 01.01.2023))



**PK:** tid (unique, not null)

**Bugs\_Tasks** (tid: int (> 0), scenario: string (600), project\_id: int (> 0), blocking\_date: date (>= 01.01.2023), unblocking\_date: date (>= blocking\_date))

**PK:** tid (unique, not null)

**FK:** Bugs\_Tasks (project\_id) -> Projects (pid)

#### ООП подход:

Tasks (tid: int (> 0), status: string (10 – unfinished), start\_date: date (>= 01.01.2023), tdescription: string (350))

Executables (tid: int (> 0), deadline: date (>= 01.01.2023))

Bugs (tid: int (> 0), scenario: string (600), project\_id: int (> 0))

Executable\_Tasks (tid: int (> 0), status: string (10 – unfinished), start\_date: date (>= 01.01.2023), tdescription: string (350), deadline: date (>= start\_date))

Bugs\_Tasks (tid: int (> 0), status: string (10 – unfinished), start\_date: date (>= 01.01.2023), tdescription: string (350), scenario: string (600), project\_id: int (> 0), blocking\_date: date (>= 01.01.2023), unblocking\_date: date (>= blocking\_date))

#### NULL подход:

Tasks (tid: int (> 0), status: string (10 – unfinished), start\_date: date (>= 01.01.2023), tdescription: string (350), deadline: date (>= start\_date), scenario: string (600), project\_id: int (> 0), blocking\_date: date (>= 01.01.2023), unblocking\_date: date (>= blocking\_date))

Избирам релационния подход, защото за удобство бих могла да си позволя да разделя атрибутите, необходими за описанието на различни типове задачи, в три таблици. С оглед на възможните транзакции към базата, обаче, този подход не би бил най-удачен, защото в част от случаи ще се обръщаме към повече от една таблица в модела, за да извлечем цялостно информацията, от която се интересуваме. Но от гледна точка на сравнително малкото количество данни в таблиците, това не би било от особено значение за бързодействието на модела. За задачите ще генерирам 3 таблици.

*Таблицы на останалите множества от същности и взаимоотношенията им*

**Companies** (cid: int (> 0), cname: string (50), caddress: string (100), cdescription: string (500))

**PK:** cid (unique, not null)

Забележка: Тъй като името на компанията ще е низ с променлива дължина, не би било удачно да е ключ, заради проблеми с индексацията на такива полета. Това налага въвеждането на ново поле за ключ, а именно cid: int (> 0).

**Teams** (team\_id: int (> 0), cid: int (> 0), tname: string (50), tdescription: string (350))

**PK:** team\_id (unique, not null)

**FK:** Teams (cid) -> Companies (cid)

Забележка: Тъй като имената на екипа и компанията ще са низове с променлива дължина, не би било удачно да сформират ключ, заради проблеми с индексацията 4 на такива полета. Това налага въвеждането на ново поле за ключ, а именно team\_id: int (>= 0).

**Projects** (project\_id: int (> 0), pname: string (50), version: string (20), release\_date: date (>= 01.01.2023), pdescription: string (350))

**PK:** project\_id (unique, not null)

Забележка: Тъй като името на проекта ще е низ с променлива дължина, не би било удачно да е ключ, заради проблеми с индексацията на такива полета. Това налага въвеждането на ново поле за ключ, а именно pid: int (>= 0).

Defines (uid: int (> 0), tid: int (> 0)) – таблицата отпада

AreFoundIn (tid: int (> 0), project\_id: int (> 0)) – таблицата отпада

AreUnitsOf (tid: int (> 0), project\_id: int (> 0)) – таблицата отпада

IsBlocked (blocking\_task\_id: int (> 0), blocked\_task\_id: int (> 0), blocking\_date: date (>= 01.01.2023), unblocking\_date: date (>= blocking\_date)) – таблицата отпада  
WorksIn (uid: int (> 0), team\_id: int (> 0)) – таблицата отпада

**WorksOn** (uid: int (> 0), tid: int (> 0))

**PK:** uid (unique, not null), tid (unique, not null)

**FK:** WorksOn (uid) -> Users (uid)

WorksOn (tid) -> Tasks (tid)

**TeamsWorkOn** (team\_id: int (> 0), project\_id: int (> 0))

**PK:** team\_id (not null), project\_id (unique, not null)

**FK:** TeamsWorkOn (team\_id) -> Teams (team\_id) TeamsWorkOn (project\_id) -> Projects (project\_id)

Забележка: дължината на полетата от тип string са проверявани от различни онлайн източници по спецификациите на SQL с цел максимална прецизност и доближаване до изискванията на задачата.

Извод

След всички преобразувания до момента стигаме до следните релации.

## Таблицу

**Users** (uid: int (>= 0), email: string (320), username: string (10), password: string (64))

**Users\_Employees** (uid: int (> 0), email: string (320), username: string (10), password: string (64), salary: double (> 0.0), emp\_address: string (100), phone: string (>= 10 && <= 13), team\_id: int (> 0))

**Users\_Amateurs** (uid: int (> 0), email: string (320), username: string (10), password: string (64), activity\_description: string (350))

**Tasks** (tid: int (> 0), status: string (10 – unfinished), start\_date: date (>= 01.01.2023), tdescription: string (350), uid: int (> 0), project\_id: int (> 0), btid: int (> 0)) Executables (tid: int (> 0), deadline: date (>= 01.01.2023))

**Bugs** (tid: int (> 0), scenario: string (600), project\_id: int (> 0), blocking\_date: date (>= 01.01.2023), unblocking\_date: date (>= blocking\_date))

**Companies** (cid: int (> 0), cname: string (50), caddress: string (100), cdescription: string (500))

**Teams** (team\_id: int (> 0), cid: int (> 0), tname: string (50), tdescription: string (350))

**Projects** (project\_id: int (> 0), pname: string (50), version: string (20), release\_date: date (>= 01.01.2023), pdescription: string (350))

**WorksOn** (uid: int (> 0), tid: int (> 0))

**TeamsWorkOn** (team\_id: int (> 0), project\_id: int (> 0))

## Първични ключове (PK)

**PK:** Users -> uid (unique, not null)

**PK:** Users\_Employees -> uid (unique, not null)

**PK:** Users\_Amateurs -> uid (unique, not null)

**PK:** Tasks -> tid (unique, not null)

**PK:** Executable\_Tasks -> tid (unique, not null)

**PK:** Bugs -> tid (unique, not null)

**PK:** Companies -> cid (unique, not null)

**PK:** Teams -> tid (unique, not null)

**PK:** Projects -> pid (unique, not null)

**PK:** WorksOn -> uid (unique, not null), tid (unique, not null)

**PK:** TeamsWorkOn -> team\_id (not null), project\_id (unique, not null)

#### Външни ключове (FK)

**FK:** Users (tname, cname) -> Teams (tname, cname)

**FK:** Tasks (uid) -> Tasks (uid)

**FK:** Tasks (pname, pversion) -> Projects (pname, version)

**FK:** Tasks (btid) -> Tasks (tid)

**FK:** Bugs (found\_in\_pname, found\_in\_pversion) -> Projects (pname, version)

**FK:** Teams (cname) -> Companies (cname)

**FK:** WorksOn (uid) -> Users (uid)

**FK:** WorksOn (tid) -> Tasks (tid)

**FK:** TeamsWorkOn (team\_id) -> Teams (team\_id)

**FK:** TeamsWorkOn (project\_id) -> Projects (project\_id)

#### Ограничения

**Users (email - \_@\_.)**

**Users (username - \_%)**

**Users (password - \_%)**

**Users\_Employees (email - \_@\_.)**

**Users\_Employees (username - \_%)**

**Users\_Employees (password - \_%)**

**Users\_Employees (salary - > 0.0)**

**Users\_Amateurs (email - \_@\_.)**

**Users\_Amateurs (username - \_%)**

**Users\_Amateurs (password - \_%)**

**Tasks (status – finished / unfinished)**

**Tasks (start\_date >= 01.01.2023)**

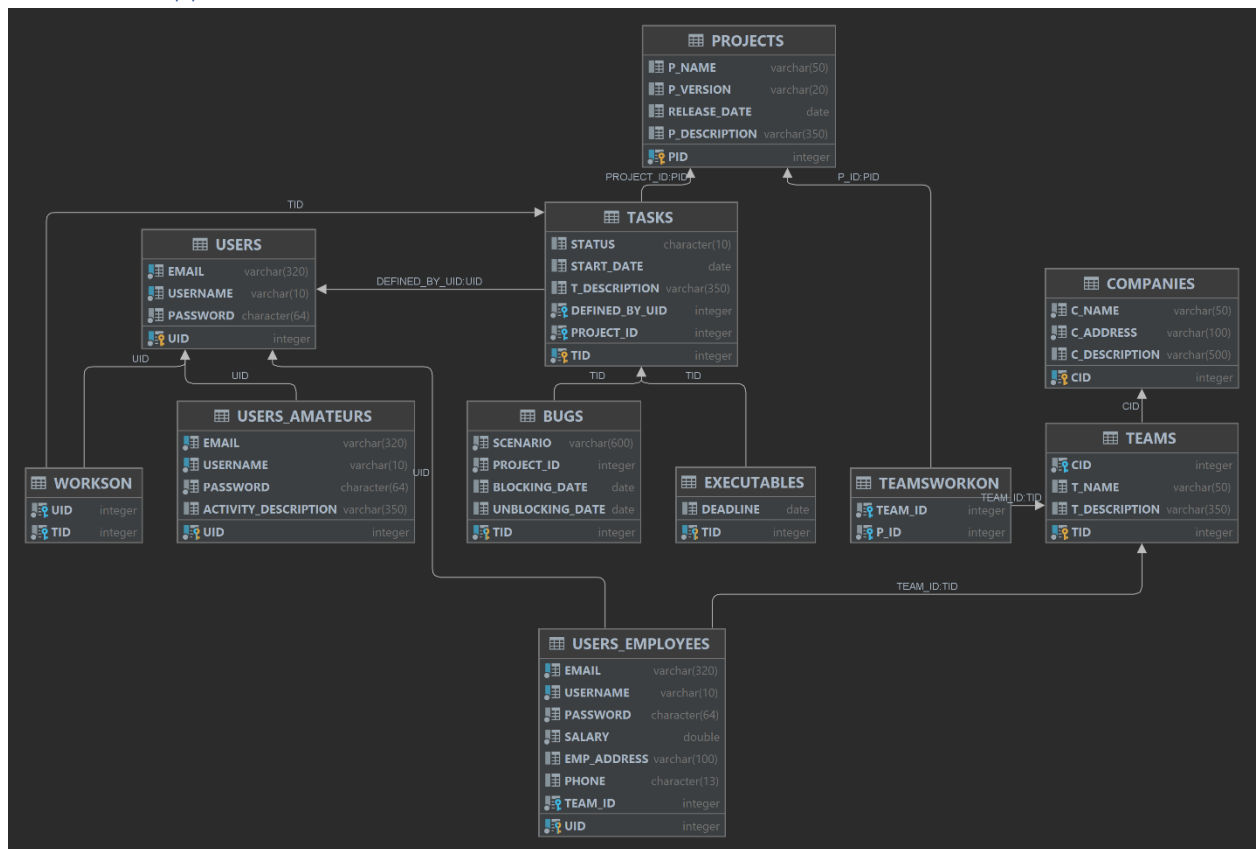
**Executable\_Tasks (deadline >= start\_date)**

**Bugs (blocking\_date >= 01.01.2023)**

**Bugs (unblocking\_date >= blocking\_date)**

**Projects (release\_date - >= 01.01.2023)**

## Схема на базат от данни



## Изгледи

## Employees view

```
CREATE VIEW "Employees" AS
SELECT FN3MI0700022.USERS_EMPLOYEES.EMAIL,
       FN3MI0700022.USERS_EMPLOYEES.USERNAME,
       FN3MI0700022.USERS_EMPLOYEES.EMP_ADDRESS,
       FN3MI0700022.USERS_EMPLOYEES.PHONE,
       FN3MI0700022.USERS_EMPLOYEES.SALARY,
       FN3MI0700022.TEAMS.T_NAME,
       FN3MI0700022.COMPANIES.C_NAME
FROM FN3MI0700022.USERS_EMPLOYEES, FN3MI0700022.TEAMS, FN3MI0700022.COMPANIES
WHERE FN3MI0700022.USERS_EMPLOYEES.TEAM_ID = FN3MI0700022.TEAMS.TID AND
      FN3MI0700022.TEAMS.CID = FN3MI0700022.COMPANIES.CID;

SELECT * FROM "Employees" WHERE C_NAME = 'Nexus Dynamics';
```

Tasks view

```
CREATE VIEW "Tasks" AS
SELECT * FROM FN3MI0700022.EXECUTABLES
WHERE DEADLINE > CURRENT_DATE;

SELECT * FROM "Tasks";
DROP VIEW "Tasks";
```

Bugs view

```
CREATE VIEW "Bugs" AS
SELECT * FROM "Bugs"
WHERE FN3MI0700022."Bugs".UNBLOCKING_DATE > FN3MI0700022."Bugs".BLOCKING_DATE;

SELECT * FROM Bugs;
INSERT INTO Bugs(SCENARIO, PROJECT_ID, BLOCKING_DATE, UNBLOCKING_DATE)
VALUES('new bug scenario', 10, '2023-04-22', '2023-05-31');
SELECT * FROM Bugs;
```

Приложение за достъп до базата от данни

main() method

- стартовата точка на приложението

```
public static void main(String[] args) {
    ProjectManagementApp db20bj = new ProjectManagementApp();
    String option = "";
    db20bj.execute(db20bj.printMenu(option));
}
```

printMenu() method

– принтира менюто и въвежда потребителя в базата данни като му дава опции, от които да си избере каква заявка иска да изпълни върху таблицата Employees

```

public String printMenu(String option) {
    Scanner input = new Scanner(System.in);
    do {
        System.out.println("""
        Welcome to Project Management DataBase!
        Choose a command to be executed!
        MENU:
        1. SELECT
        2. INSERT
        3. DELETE
        4. EXIT
        """);
        option = input.next();
    } while (!option.equals("SELECT") && !option.equals("INSERT")
        && !option.equals("UPDATE") && !option.equals("DELETE")
        && !option.equals("EXIT"));

    return option;
}

```

execute()

– извиква и изпълнява конкретен метод, който се грижи за даден тип заявки

```

public void execute(String option) {
    String statement = ""; this.openConnection();
    switch (option) {
        case "SELECT": {
            statement = option + " * FROM FN3MI0700022.USERS";
            this.select(statement, column: 4);
            break;
        } case "INSERT": {
            Scanner input = new Scanner(System.in);
            String mail = input.next();
            String username = input.next();
            String pass = input.next();

            statement = option + " INTO FN3MI0700022.USERS (EMAIL, USERNAME, PASSWORD)"
                + " VALUES ('" + mail + "','" +
                username + "','" + pass + "')";
            this.insert(statement);
        } case "DELETE": {
            Scanner input = new Scanner(System.in);
            String username = input.next();
            statement = option + " FROM FN3MI0700022.USERS WHERE FN3MI0700022.USERS.USERNAME = '"
                + username + "'";
            this.delete(statement);
        } case "EXIT": default: System.exit(status: 0);
    } this.closeConnection();
}

```

Конзола с резултатните множества от заявките

*SELECT*

– показва всички записи в таблицата Employees с всичките им атрибути и съответните им стойности

```

Welcome to Project Management DataBase!
Choose a command to be executed!
MENU:
1. SELECT
2. INSERT
3. DELETE
4. EXIT

SELECT
1      fakeemail1@yahoo.com      coolguy56      abcd1234
2      wtpfakeemail@gmail.com    hotsauce      qwertyuiop
3      notreal@email.co.uk      lilypad44      password123
4      dummyaddress@outlook.com  speedy423      ilovemom1
5      fictitiousemail@yahoo.com  cloudberry     123456789

```



### INSERT INTO

– добавя запис в таблицата Employees като потребителя трябва ръчно да въведе стойност в конзолата на приложението за всяко необходимо поле

	UID	EMAIL	USERNAME	PASSWORD
6	6	tempuser9990@gmail.com	jollyjane	letmein521
7	7	phantomemail@protonmail.com	boldbob	mydogspot29
8	8	unrealaddress@yahoo.com	crazykyle	8675309jk
9	9	fictionalaccount@gmail.com	happyharry	flymetothemoon
10	10	unrealuser23@outlook.com	zippyzoie	abcde12345
11	21	someone@gmail.com	someone	myPass!123

```
ProjectManagementApp x
↑ Choose a command to be executed!
↓ MENU:
1. SELECT
2. INSERT
3. DELETE
4. EXIT

INSERT
someone@gmail.com
someone
myPass!123
INSERT INTO FN3MI0700022.USERS (EMAIL, USERNAME, PASSWORD) VALUES ('someone@gmail.com','someone','myPass!123')
Successfully inserted!
```

### DELETE

– изтрива запис за служител от таблицата Employees по подадено име на служителя, ако такъв съществува

```
Welcome to Project Management DataBase!
Choose a command to be executed!
MENU:
1. SELECT
2. INSERT
3. DELETE
4. EXIT

DELETE
someone
DELETE FROM FN3MI0700022.USERS WHERE FN3MI0700022.USERS.USERNAME = 'someone'
Successfully deleted!
```

### EXIT

– излиза от приложението и терминира работата на конзолата

```
Welcome to Project Management DataBase!  
Choose a command to be executed!  
MENU:  
1. SELECT  
2. INSERT  
3. DELETE  
4. EXIT
```

EXIT

Process finished with exit code 0

|