

Диаграммы приложения

```

graph LR
    subgraph Client ["Client<br/>«device»<br/>PC running WinBox"]
        FXML[FXML]
        ClientApp[Client Application]
        FXML -.-> ClientApp
    end

    subgraph Server ["Server<br/>«device»<br/>PC running WinBox"]
        Maven[Maven]
        AppServer[Application Server]
        Console[Console]
        Maven --> AppServer
        AppServer --> Console
    end

    subgraph MySQL_server ["MySQL server<br/>«device»<br/>PC running WinBox"]
        MySQL[MySQL server]
    end

    ClientApp -- "TCP/IP" --> AppServer
    AppServer -- "JDBC-driver" --> MySQL
  
```

```

sequenceDiagram
    participant User as Пользователь
    participant Controller as Контроль
    participant Service as Сервис
    participant DB as База данных

    User->>Controller: запрос доступных витков
    Controller->>Service: передача данных
    Service->>DB: запрос данных
    DB-->>Service: ответ
    Service->>Controller: передача данных
    Controller-->>User: вывод доступных витков

    User->>Controller: ввод данных витков
    Controller->>Service: запрос сокрытия подвешивания
    Service->>DB: отправка данных в БД
    DB-->>Service: обработка
    Service->>Controller: передача обработанных данных
    Controller-->>User: отправка сокрытых данных

    User->>Controller: редактирование витков
    Controller->>Service: передача новых данных
    Service->>DB: отправка новых данных в БД
    DB-->>Service: обработка
    Service->>Controller: передача новых данных
    Controller-->>User: сокрытие от публики
  
```

The diagram illustrates a three-tier architecture. The **Client** tier contains **ControllerFXML**, **FXML**, and **GUI**. The **Server** tier contains **Business logic**, **Control classes**, and **Database control classes**. The **Database** tier contains the **MySQL database**. Components are represented by rectangles with a small document icon in the top right corner. Solid arrows indicate dependencies: **ControllerFXML** depends on **FXML** and **GUI**; **FXML** depends on **GUI**; **ControllerFXML** depends on **JavaFX**; **JavaFX** depends on **Workstation**; **Workstation** depends on **Java files**; **Java files** depends on **Classes**; **Business logic** depends on **Control classes**; **Control classes** depends on **Database control classes**; **Database control classes** depends on the **MySQL database**. Dashed arrows indicate associations: **ControllerFXML** is associated with **FXML**; **FXML** is associated with **GUI**; **ControllerFXML** is associated with **JavaFX**; **JavaFX** is associated with **Workstation**; **Workstation** is associated with **Java files**; **Java files** is associated with **Classes**; **Business logic** is associated with **Control classes**; **Control classes** is associated with **Database control classes**; **Database control classes** is associated with the **MySQL database**. A **TCP/IP** connection is shown between the **Client** and **Server** tiers, and a **JDBC-driver** connection is shown between the **Server** and **Database** tiers.

```

    erDiagram
        users ||--o{ employees : "manages"
        users ||--o{ admins : "manages"
        users ||--o{ assets : "manages"
        users ||--o{ depreciation : "manages"
        employees ||--o{ assets : "owns"
        admins ||--o{ assets : "owns"
        assets ||--o{ depreciation : "depreciates"

        users {
            string id PK
            string login
            string password
        }
        employees {
            string id PK
            string nameEmployee
            string surname
            string department
            string idAssets
            int int
        }
        admins {
            string id PK
            string login
            string password
        }
        assets {
            string id PK
            string name
            int price
            int termofUse
            int incomeMoney
        }
        depreciation {
            string id PK
            string name
            double yearPrice
            double monthPercent
            double payBack
            double nerability
            double totalBenefit
        }
  
```

The diagram illustrates the relationships between five tables: **users**, **employees**, **admins**, **assets**, and **depreciation**. Each table has a primary key (PK) and specific attributes. The relationships are as follows:

- users** (PK: id) is connected to **employees** (PK: id), **admins** (PK: id), **assets** (PK: id), and **depreciation** (PK: id) via a 1-to-many relationship (indicated by a line with a crow's foot notation).
- employees** (PK: id) is connected to **assets** (PK: id) via a 1-to-many relationship.
- admins** (PK: id) is connected to **assets** (PK: id) via a 1-to-many relationship.
- assets** (PK: id) is connected to **depreciation** (PK: id) via a 1-to-many relationship.

The attributes for each table are:

- users**: id (PK), login, password.
- employees**: id (PK), nameEmployee, surname, department, idAssets, int.
- admins**: id (PK), login, password.
- assets**: id (PK), name, price, termofUse, incomeMoney.
- depreciation**: id (PK), name, yearPrice, monthPercent, payBack, nerability, totalBenefit.

```
graph TD; Start((Авторизация)) -.->|Ввести логин и пароль| Step1[Логин и пароль введены]; Step1 -.->|Отправить данные| Step2[Данные отправлены на проверку]; Step2 -.->|Вывести ошибку| Step3[Некорректные символы]; Step3 -.->|Ввести данные повторно| Step1; Step2 -.->|Проверить на соответствие| Step4[Проверка на соответствие]; Step4 -.->|Вывести ошибку| Step2; Step4 -.->|Сравнить с имеющимися данными| Step5[Данные проверены]; Step5 -.-> End((Авторизованный пользователь)); Step6[Данные отсутствуют] -.->|Вывести ошибку| Step4; Step6 -.->|Ввести данные повторно| Step1;
```

[illegible]

```
graph TD; Start([Начало]) --> Step1[Соединение с сервером]; Step1 --> Step2[Соединение с БД]; Step2 --> Decision{Данные есть в БД?}; Decision -- Да --> Step3[Получение наименования актива]; Decision -- Нет --> Step4[/Данные не получены/]; Step3 --> Step5[Получение стоимости актива]; Step5 --> Step6[Получение количества актива]; Step6 --> Step7[Получение срока годности]; Step7 --> Step8[Расчет порождаемости]; Step8 --> Step9[Расчет годовых издержек]; Step9 --> Step10[Расчет рентабельности]; Step10 --> Step11[Расчет общей выгоды]; Step11 --> Step12[Отправка данных в БД]; Step12 --> End([Конец]); Step4 --> End;
```

					ГУИР КП 1-40 05 01 012 Д7				
					Управление ИТ-активами предприятия и его программная поддержка	Лит.		Масса	Масштаб
Изм.	Лист	№ докум.	Подп.	Дата					
Разраб.	Ерёменко					Г			
Пров.	Унучек								
						Лист		Листов 1	
Н.контр.	Унучек					ПИКС зр.914301			
	Хорошко								