Online Private Tutors Finder System

Admin
```
+ getParents() : List<Parent>
+ addBook(name : String, author : String, yearPublished : int, size : int) : void
+ addTutor(username : String, password : String) : User
+ Admin(username : String, password : String, db : DataBase)
```

User
```
# db : DataBase
- password : String
- username : String

+ getUsername() : String
+ checkPassword(p : String) : boolean
- User(username : String, password : String, db : DataBase)
```

DataBase
```
+ books : List<Book>
- parents : List<Parent>
- tutors : List<Tutor>
- users : HashMap<String, User>

+ getParents() : List<Parent>
+ getTutors() : List<Tutor>
+ register(username : String, password : String, userType : String) : User
- login(username : String, password : String) : User
```

Book
```
+ size : int
+ yearPublished : int
+ author : String
+ name : String

+ Book(name : String, author : String, yearPublished : int, size : int)
```

Parent
```
- bookedTutors : List<Tutor>
- lastFilter : Filter

+ printListOfTutors(list : List<Tutor>) : void
+ bookTutor(tutor : Tutor) : void
+ rateTutor(tutor : Tutor, rate : int) : void
+ sendRequest(tutor : Tutor, request : String) : void
+ searchBook(name : String) : List<Book>
+ createFilter(str : String) : Filter
+ filter(filter : Filter) : List<Tutor>
+ getLastFilter() : Filter
+ Parent(username : String, password : String, db : DataBase)
```

<<interface>> Filter
```
getTutors(tutors : List<Tutor>) : List<Tutor>
```

Tutor
```
- requests : List<String>
- name : String
- rating : int

+ addRating(rate : int) : void
+ setRating(rating : int) : void
+ getName() : String
+ setName(name : String) : void
+ getRequests() : List<String>
+ addRequest(request : String) : void
+ getRating() : int
+ Tutor(username : String, password : String, db : DataBase)
```

FilterByRating
```
- incr : boolean (readOnly)
- to : int (readOnly)
- from : int (readOnly)

+ getTutors(tutors : List<Tutor>) : List<Tutor>
+ FilterByRating(from : int, to : int, incr : boolean)
+ FilterByRating()
```

FilterByName
```
- name : String (readOnly)

+ getTutors(tutors : List<Tutor>) : List<Tutor>
+ FilterByName(name : String)
```
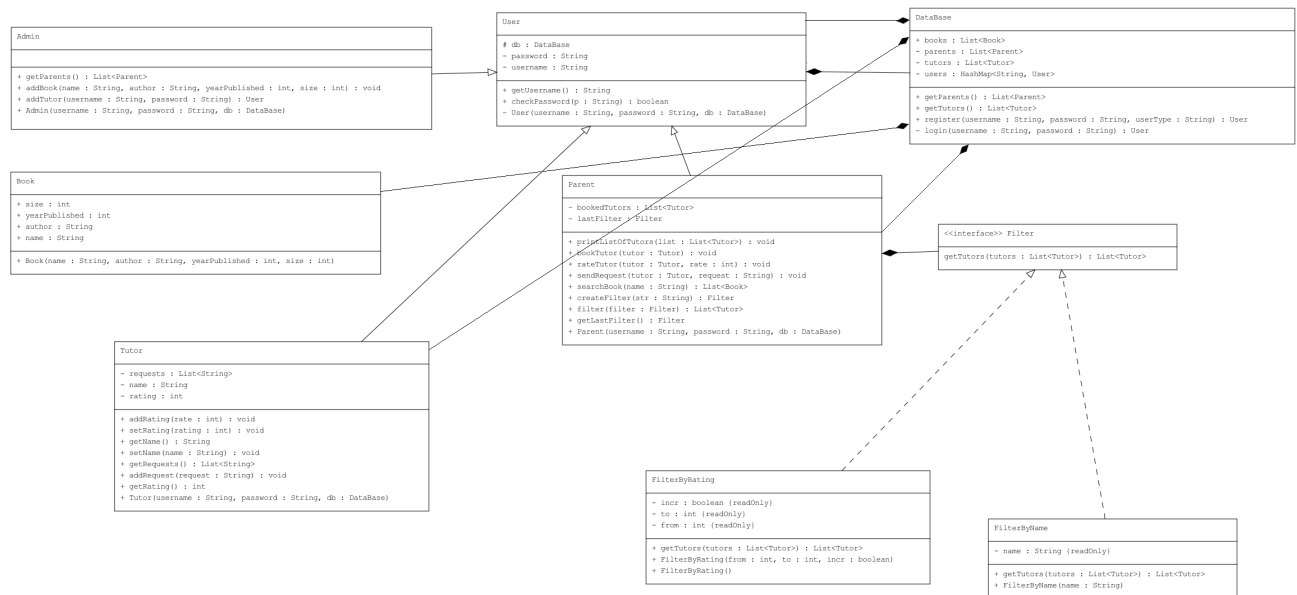
Report

Our team implemented Tutors Finding System to help find tuition teachers from nearby locations. The system has users such as Tutor, Parent, and Admin which are derived from User superclass. Each type of user has its specific functionality. For example, Parent can filter tutors by `filter(...)`.

To operate with the pool of users and other objects, it was suggested to create class "DataBase" which will provide an interface for interacting with pools of objects. One of the responsibilities of DataBase is to register users which is done by User `register(...)`.

For both filtering tutors and registering users, it was decided to use fabric methods. The reason is that it provides a convenient way to extend internals if, for example, we decide to add a new type of filter or a new type of user. `register(...)` method, based on the arguments, creates and returns the corresponding user type. Similarly, createFilter(…) creates and returns a Filter class with specified filter criteria which will be used by `filter(...)`. It is most likely, that there will be demand for new filters, so it can be easily done by adding a filter to the `createFilter(...)`.

Overall, the system provides functionality for registering users(`register`), booking(`bookTutor`), rating(`rateTutor`), finding and filtering tutors(`filter`), and other functionality needed.