Feature Gating Module

Design a feature gating module which evaluates whether the user is allowed to access a particular feature or not depending on conditional expression evaluated against user attributes.

Feature

Feature can be either a functionality or a privilege within the app.

Examples of features could be

1. Exclusive access to certain sets of categories.
2. Privileges like Same day delivery, 30 day returns etc.

User Attributes

User Attributes can be defined as the properties (implicit, explicit or derived) or traits of the user. In this context, user attributes could be

Sample set of user attributes
Note - The attribute can be added at a later point of time.
User can have many attributes, this could be **sparse** i.e. some user may not have a well defined attribute (Ex: We might not have location captured for the user)

Each attribute of the user can be associated with a particular Data Type (`number, string, boolean`)

Attribute Name	**Data Type**	**Comments**
gender	string	Male, Female, Unknown
age	number	1 .. 100
salary	number	10000.0
height	number	In centimeter
is_affluent	boolean	true/false
city	string	Delhi, dubai
spends	number	123.33 , (total spend per

		mon)
latitude	number	12.74
longitude	number	77.723

Conditional Expression

In this context, condition is basically an infix expression to decide whether the user is allowed to use a particular feature or not

Ex: `(age > 25 AND gender == "male") OR (past_order_amount > 10000)`
“Read as”
If the age is greater than 25 and gender is male or if the total purchase in the past exceeds 10,000

If this condition evaluates to true, user will be allowed to use the feature

Input to the module

```
boolean isAllowed(String featureName, String conditionalExpression, Map<String, Object> userAttributes);
```

Function takes a name of feature, a conditional Expression, and user attributes defined as a complex map and evaluates the expression and returns a boolean.

Examples

Expression	**Context Map**	**Result**
`(age > 25 && gender == male) || past_order_amount > 10,000)`	`{ “age” : 24, “gender” : “female”, “past_order_amount” : 9000 }`	`false`

`(age > 25 && gender == male) || past_order_amount > 10,000)`	`{ “age” : 25, “gender” : “male”, “past_order_amount”: 9000 }`	`true`
`(age > 25 && gender == male) || past_order_amount > 10,000)`	`{ “age” : 24, “gender” : “female”, “past_order_amount”: 11000 }`	`true`

Functional requirements

1. `conditionalExpression` should be able to support any infix expression
2. Supported operators `>, >=, <=, <, ==, !=, BETWEEN, ALLOF, NONEOF`
3. Data types supported `numbers, string, boolean`, there must be validation between data type of attribute and the operators that can be applied.
4. The module should be extensible to allow new attributes and new operators without major changes to the code.

Notes

1. Follow standard OOPs concept and best practices in the industry.
2. List the user attributes, operators and data types supported.
3. Write unit tests to test the behaviour.
4. Mention assumptions made if any. Please add appropriate comments when stubbing any complex logic.