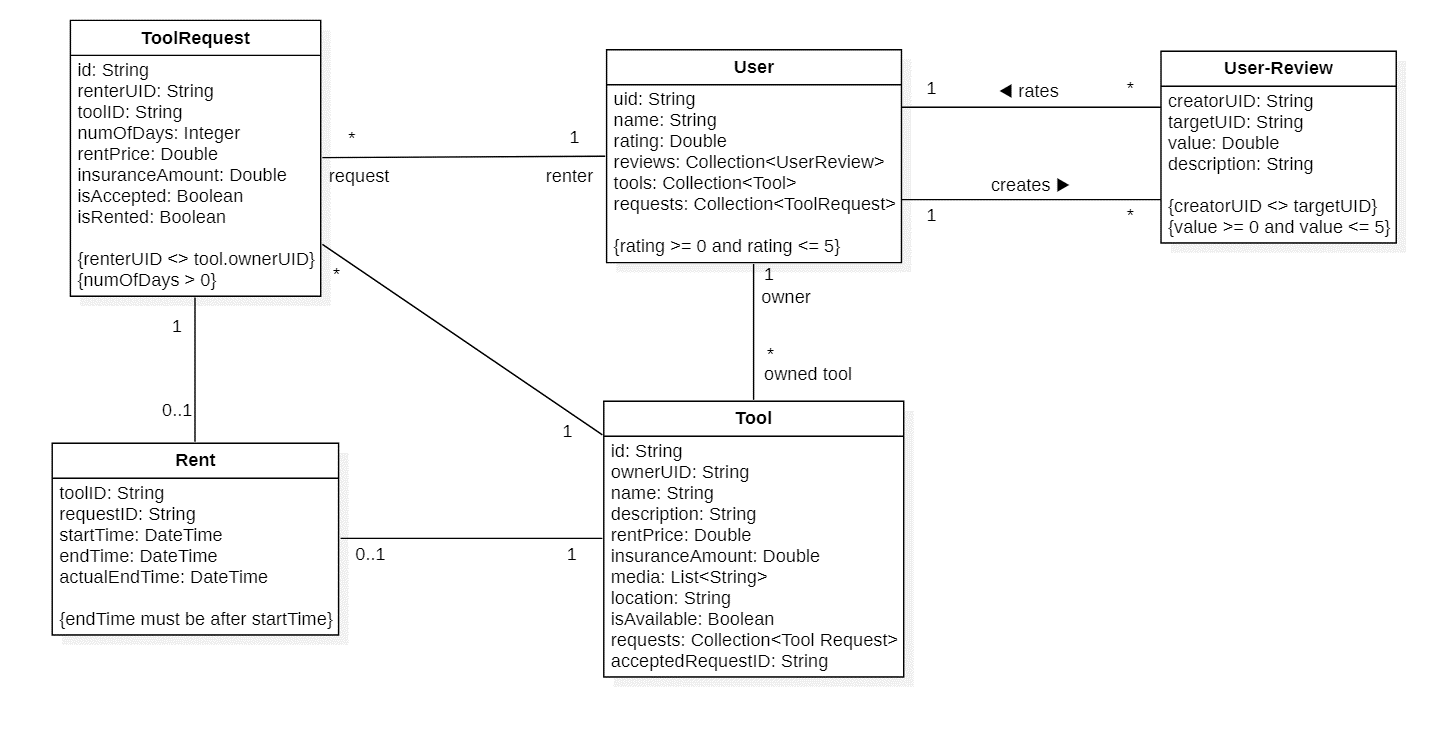
Constraints



**Context** User **inv:**

self.uid == self.tool.ownerUID

**Context** User **inv:**

self.uid == self.request.renterID

**Context** User **inv:**

self.uid <> Self.request.tool.ownerUID

**Context** ToolsRequest **inv:**

Self.isAccepted == true **implies** self.id == self.tool.acceptedRequestID

**Context** ToolsRequest **inv:**

self.renterUID <> self.tool.ownerUID

**Context** Rent **inv**:

self.toolID == self.request.toolID

**Context** Rent **inv**:

self.tool == self.request.tool

# System Operations

---------------------------------------------

**Context** User:: createPost(name: String, description: String, rentPrice: Double, insuranceAmount: Double, media: List<String>, location: String)

**Pre:**

* name.length > 0
* rentPrice > 0
* insuranceAmount > 0
* location.length > 0

**Post:**

* a new Tool object will be created.
* the user will be linked with the Tool object.
* if the operation succeeded the Tool object’s document-reference will be returned.
* otherwise (if an error occurred) a string indicating the error will be returned.

---------------------------------------

**Context** Tool::Edit({name: String, description: String, rentPrice: Double, insuranceAmount: Double, media: List<String>, location: String})

**Pre:**

* name.length > 0
* rentPrice > 0
* insuranceAmount > 0
* location.length > 0

**Post:**

* the Tool object will be edited.
* if the operation succeeded the Tool object’s document-reference will be returned.
* otherwise (if an error occurred) a string indicating the error will be returned.

**--------------------------------**

**Context** Tool::acceptRequest(toolRequestID)

**Pre:**

* Tool.requests should include toolRequestID.
* Tool.isAvailable == true

**Post:**

* The ToolRequest will be accepted
* The ToolRequest and tool will be linked
* Tool.isAvailable will become false
* A chat room will be created
* The Tool owner will be linked with the chat room
* The ToolRequest sender (renter) will be linked with the chat room
* The ToolRequest sender (renter) will be notified
* if an error occurred a string indicating the error will be returned.

---------------------------------------------------------

**Context** Tool :: startRent()

**Pre:**

* The tool should have an accepted tool-request (i.e., Tool.isAvailable == false)

**Post:**

* A new Rent object, rent, will be created as follows:

rent.ID will become unique identifier

rent.toolID will become tool.ID

rent.requestID will become tool.acceptedRequestID

rent.startTime will become DateTime.now()

* tool.currentRentID will become rent.ID
* The Rent object will be linked with the Tool
* The Rent object will be linked with the ToolRequest
* if the operation succeeded the Rent object’s document-reference will be returned.
* otherwise (if an error occurred) a string indicating the error will be returned.

---------------------------------------------------------

**Context** Tool:: addRequest(renterUID: String, toolID: String, numOfDays: Integer, rentPrice: Double, insuranceAmount: Double)

**Pre:**

* renterUID is a valid user ID
* toolID is a valid tool ID
* toolID doesn’t reference a tool where tool.ownerID == renterID (i.e., a user can’t request his/her own tool)
* numOfDays > 0
* rentPrice > 0
* insuranceAmount > 0

**Post:**

* a new ToolRequest object will be created.
* The object will be linked with the renter
* The object will be linked with the Tool object
* if the operation succeeded the ToolRequest object’s document-reference will be returned.
* otherwise (if an error occurred) a string indicating the error will be returned.

---------------------------------------------------------

**Context** ToolRequest:: edit(numOfDays: Integer, rentPrice: Double, insuranceAmount: Double)

**Pre:**

* numOfDays > 0
* rentPrice > 0
* insuranceAmount > 0

**Post:**

* the ToolRequest object will be edited.
* if the operation succeeded the ToolRequest object’s document-reference will be returned.
* otherwise (if an error occurred) a string indicating the error will be returned.

---------------------------------------------------------

**Context** Tool:: endRent()

**Pre:**

* currentRentID <> null (i.e., there should be a rent in progress)

**Post:**

* tool.currentRent.endTime will become DateTime.now()
* tool.isAvailable will become true
* tool.currentRentID will become null
* tool.acceptedRequest.isArchived will become true
* tool.acceptedRequestID will become null
* if an error occurred a string indicating the error will be returned.

---------------------------------------------------------

**Context** User:: addReview()

**Pre:**

* user.hadAPreviousRent(target) == true

**Post:**

* a new Review object, review, will be created
* review will be linked to user
* review will be linked to target
* if the operation succeeded the Review object’s document-reference will be returned.
* otherwise (if an error occurred) a string indicating the error will be returned.

---------------------------------------------------------

**Context** User:: ban()

**Pre:**

* initiator.isAdmin == true (i.e., the initiator must be an administrator)

**Post:**

* user.IDNumber will be added to the banned list
* user.emailAddress will be added to the banned list
* any unarchived tool-requests from the user will be deleted