# Rate-Book

Requirements

1. **Introduction and Context**

***Project*** *–An application to provide rating to miscellaneous mundane things.*

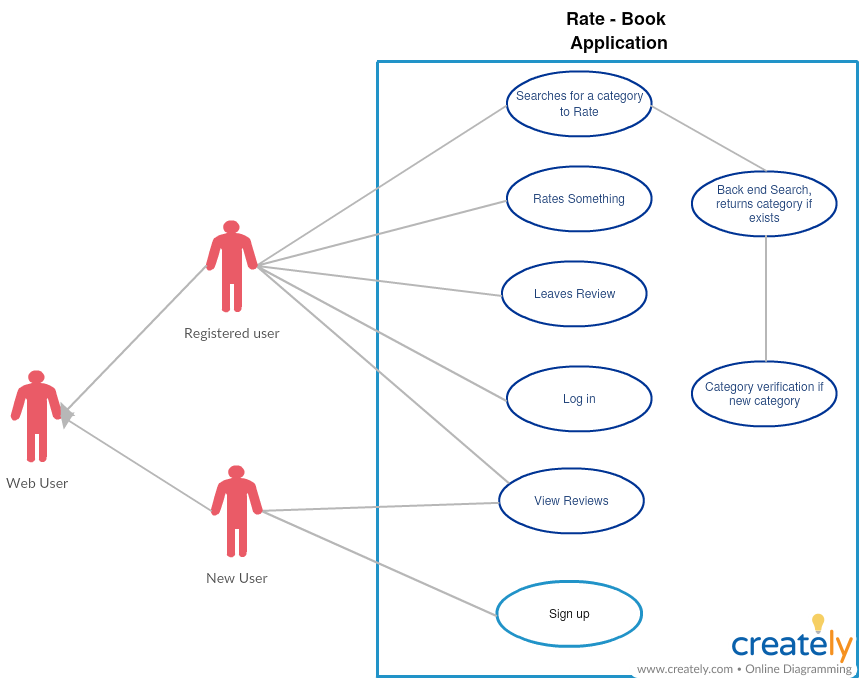
***Aim*** *- This web-application will let people rate everything/anything possible. Ex: Universities, Professors, Food, Products, Cities, courses, services and even people. Like Quora lets people ask any Question/Query and other people give their opinions, similarly this web-app will let people rate and share their experiences with anything they want.*

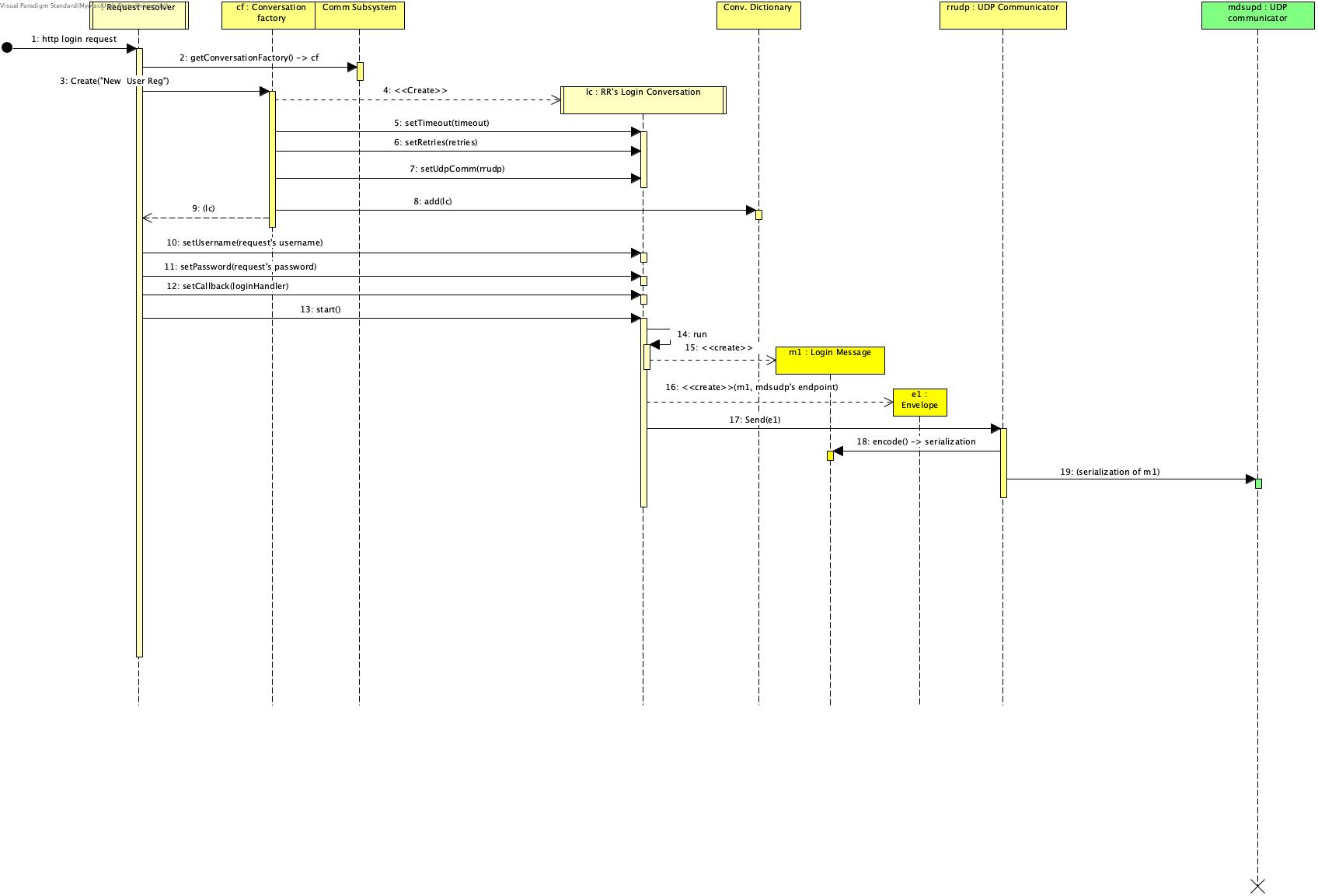
1. **Actors and their Goals**

***Actors****:*

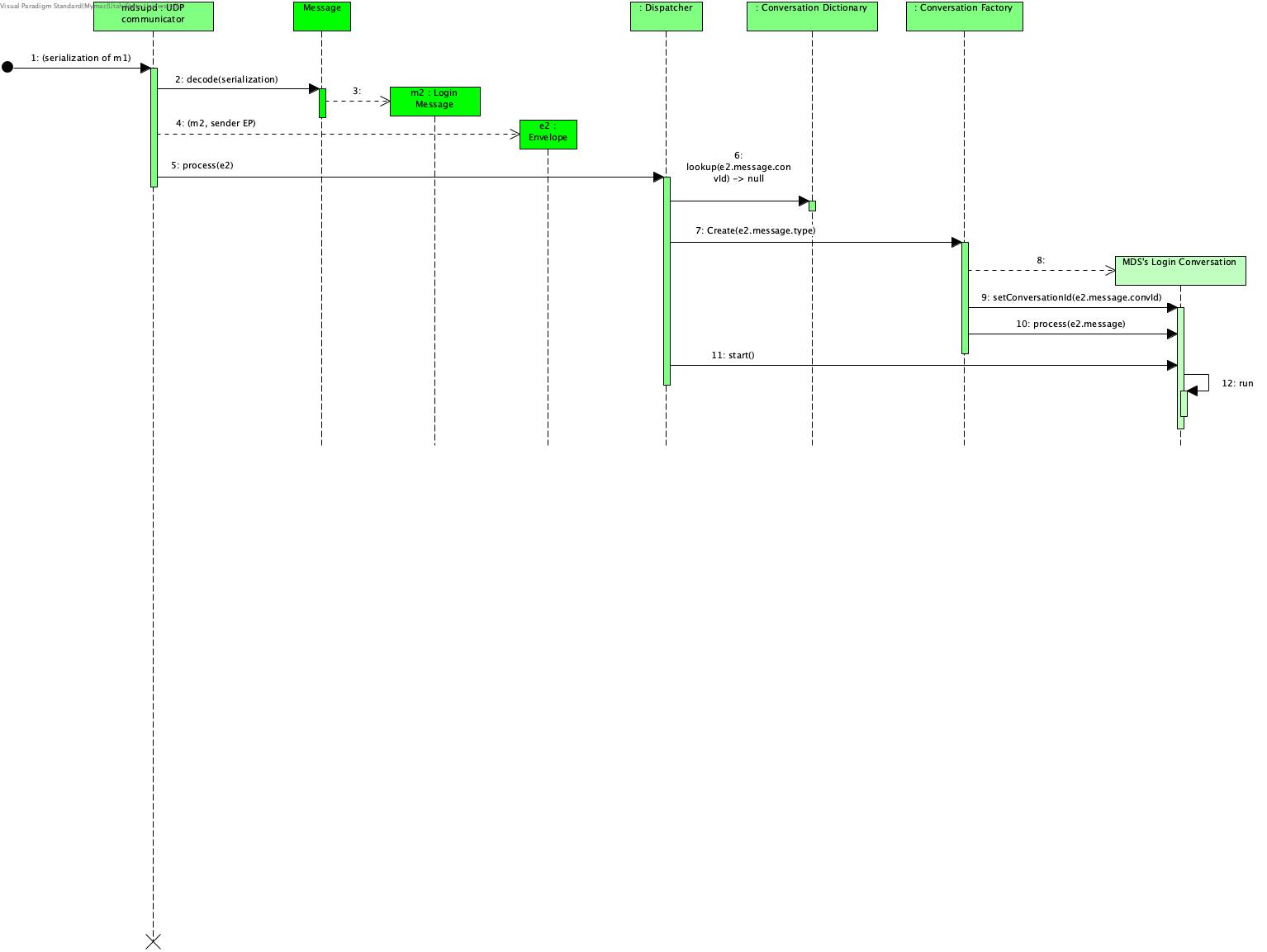
* *Web Users*

***Goals****:*

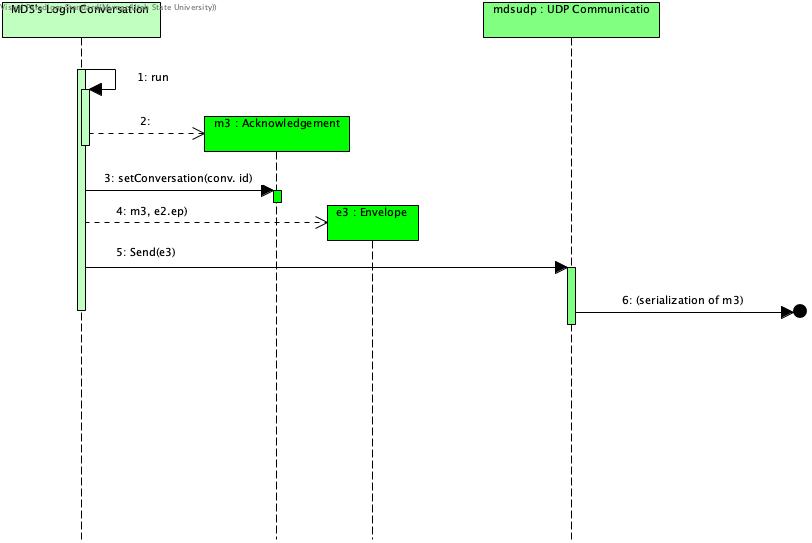
* 1. ***Log in*** *- Users will have to login to rate something.*
  2. ***Rate Something*** *- Registered users will rate something they bought or got to know about.*
  3. ***Search rating category*** *- Users can search a category of product to want to know about or share their views about.*
  4. ***Leave Reviews*** *- Only registered users can leave reviews and rate an item.*
  5. ***View Reviews*** *- All the web users can view reviews and rating about anything.*
  6. *****Sign Up*** *- New web users can sign up if they want to rate anything and share their experiences about an item.*

**

*Login Stage 1*

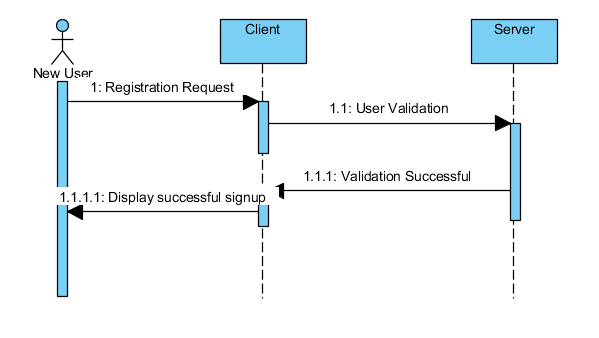
**

*Login Stage 2*

**

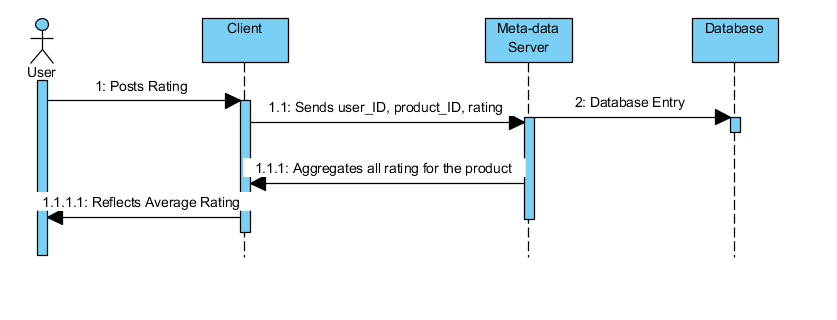
*Login Stage 3*

1. **Functional Requirements**
2. ***Client*** *– Client would be made on HTML or javafx, it would contain the features of signing up a new user and logging in an already existing user. Once a user is logged in, he will encounter a search box to search for different items. The box will provide immediate results in the drag down if the searched item exists, moreover a button to add an item in the system appears if it doesn’t already exist.*



1. *There will be 3 servers to manage the application*

* ***Searching Server****: Searching server manages all the search related activities by user, it uses a memory cache to display search results at a faster rate.*
* ***Meta-Data Server****: This server deals with user management requests and response. It gives user privilege to view ratings and reviews. Note that It won’t let user post anything until It is logged in.*



* ***Synchronization server*** *– This server is responsible for bringing in sync all the updates and manages to reflect changes made by a user over an item or a newly added category to another user.*

1. ***Memory Cache*** *– Cache for database servers to cache hot feeds and users. Will use an off-the-shelf solution like Mem-cache that can store the whole feed objects. When the cache is full and we want to replace a feed with a newer/hotter tweet, how would we choose? LRU can be a reasonable policy for our system. Under this policy, we discard the least recently viewed feed first.*
2. *Users should be able to post new comments about a product or thing*
3. *A user should be able to follow a product or thing*
4. *Users should be able to mark a product or thing as favorites*
5. *The service should be able to create and display a user’s timeline consisting of the top comments about a product or thing that the user follows from all the people*
6. *Comments can contain photos and videos*
7. **Non-functional Requirements**

* *Our service needs to be highly available*
* *Acceptable latency of the system is 2s for timeline generation*
* *Consistency can take a hit (in the interest of availability); if a user doesn’t see comment for a while, it should be fine.*
* *The programming language to build this system will be java and IntelliJ as Integrated Development Environment.*
* *Provide an accessible and usable User Interface to someone who has basic computer knowledge.*
* *Have a clean object oriented design allowing good maintainability*

1. **Future Features**

*If we go with 80-20 rule, that is 20% of tweets generating 80% of reading traffic which means that certain tweets are so popular that a majority of people read them. This dictates that we can try to cache 20% of daily read volume from each shard.*