Reddit-like Engine with REST Architecture in Go

This project implements a simplified Reddit-like engine and REST APIs using Go and the *Proto Actor* framework. The API routes

Demo

The high-level overview and demo can be found on this link : Demo Video

Overview

The project consists of six main components:

1. `main.go`:The entry point of the system.
2. `messages.go`: Defines the message structures used for communication between actors.
3. `models.go`: Contains the data models for users, subreddits, posts, and comments.
4. `engine.go`: Implements the core logic of the Reddit-like system.
5. `client.go`: Contains the REST API requestS handler functions.

Implemented Features

1. Register User.
2. Create, Join & Leave a SubReddit.
3. Post to any SubReddit, being a subscriber is not necessary, just like real world.
4. Comment on Posts and reply/comment on comments.
5. Posts can be Upvoted and Downvoted.
6. Whenever a post is created, it gets 1 upvote by default (Author's).
7. Karma Rules.-
    1. +1 Karma per post.
    2. Karma from posted content = Total Upvotes - Total Downvotes
    3. +1 Karma after commenting on a Post.
    4. -1 Karma for each downvote on the posts made.
8. Send direct message to an user.
9. Reply to direct message sent by another user.
10. Get feed for joined SubReddits.

Below is an expanded README section containing instructions on using the REST API calls to interact with the engine. Each route is listed with the JSON it expects and the typical responses.

API Endpoints

Initialize Server

To begin the server, use the following command. By default, the server runs on `http://localhost:8080`. This can be modified by changing the server in *Recieve* method in `engine.go`.

```
go run .
```

Register User

- **Route**: `/register`
- **Method**: `POST`
- **Request JSON**:

```
{
  "username": "exampleUser"
}
```

Create Subreddit

- **Route**: `/createsub`
- **Method**: `POST`
- **Request JSON**:

```
{
  "creator": "exampleUser",
  "name": "exampleSubreddit"
}
```

Join Subreddit

- **Route**: `/joinsub`
- **Method**: `POST`
- **Request JSON**:

```
{
  "subName": "exampleSubreddit",
  "user": "exampleUser"
}
```

Leave Subreddit

- **Route**: `/leavesub`
- **Method**: `POST`
- **Request JSON**:

```
{
  "subName": "exampleSubreddit",
```

```
  "user": "exampleUser"
}
```

Create Post

- **Route**: `/createpost`
- **Method**: `POST`
- **Request JSON**:

```
{
  "subredditName": "exampleSubreddit",
  "author": "exampleUser",
  "title": "Example Title",
  "content": "Example content of the post."
}
```

Create Comment

- **Route**: `/createcomment`
- **Method**: `POST`
- **Request JSON**:

```
{
  "postID": "Post [n]",
  "parentID": "Post [n]", //or 'Comment [n]' in case of heirarchical comments
  "author": "exampleUser",
  "content": "This is a comment."
}
```

Vote on Post

- **Route**: `/vote`
- **Method**: `POST`
- **Request JSON**:

```
{
   "userID": "exampleUser",
   "postID": "Post [n]",
   "isUpvote": true // false for downvote
 }
```

Get User Feed

- **Route**: `/getfeed`

- **Method**: `Get`
- **Request JSON**:

```
{
  "username": "[username]"
}
```

- **Response**
  - If successful, returns a list of posts from subscribed subreddits.
  - On error or if no posts are available, an appropriate message is returned.

Send Direct Message

- **Route:** `/sendmessage`
- **Method:** `POST`
- **Request JSON:**

```
{
  “from”: “[sender]”,
  “to”: “[recipient]”,
  “content”: “[Content]”
}
```

Summarize Session

- **Route:** `/summarize`
- **Method:** `GET`
- **Response**
  - User-wise Actions.
  - Subreddit-wise Members, Posts & Comments and their stats.
  - Simulation stats like users, actions, karma, total comments, votes etc.

Engine Methods

The Engine actor handles the core functionality of the Reddit-like system. Here's an overview of its main methods:

`Receive(context actor.Context)` : Handles incoming messages and routes them to appropriate methods.

`registerUser(username string)` : Creates a new user with the given username.

`createSubreddit(name, creator string)` : Creates a new subreddit with the given name and creator.

`joinSubreddit(subredditName, username string)` : Adds a user to a subreddit's member list.

`leaveSubreddit(subredditName, username string)` : Removes a user from a subreddit's member list.

`createPost(postID, subredditName, author, title, content string)` : Creates a new post in a specified subreddit.

`createComment(postID, parentID, commentID, author, content string)` : Adds a comment to a post or as a reply to another comment.

`vote(postID, userID string, isUpvote bool)` : Records a user's vote (upvote or downvote) on a post.

`sendDirectMessage(from, to, content string)` : Sends a direct message from one user to another.

`getFeed(username string)` : Generates a feed of posts from subreddits the user is subscribed to.

`getSimulationStats()` : Prints out statistics about users, subreddits, and posts.

Authors

1. Nirvisha Soni : 47638268
2. Neel Malwatkar : 68517665