README

==Welcome to Backchannel app

The backchannel app is a functionally improved version of Live Question app we used in class. The original Live Question app provides an open view of posts and comments, while having some important limitations. There are several improvements:

The post function of this backchannel app is improved by an added user-login-system, which can not only limit guest viewers from posting without login, but also provide convenience for voting system.

The voting system and post activity made it possible for the posts displaying an order that is highly relevant with our course progress.

The search function can help users to find their interested topics.

Users can also comment to posts to answer questions or to be involved in discussion, login users can also vote for replies.

The admin-user system can help administrator manage posts, users and grading.

==User Groups and Access Limitation

The users are divided into three groups, guests, users and administrators, according to their access limitations.

The user without an account of the app stored in database is a guest who can only view or search posts. They cannot post, edit, comment or vote, and don't have any right to view or edit other user information. Any guest can register to be a logged-in user with a unique username and password.

A logged-in user can post a new post, comment to posts created by any logged-in user, and vote posts or replies that are **not** create by him/herself. Furthermore, a logged-in user can also destroy the post created by him/herself, and can also destroy comment by him/herself.

The third group is administrator group, they have all access authorization that logged-in users have, and only they can create other admin accounts. They can delete posts and users. They can also view reports on post activity, including number of votes for each post.

==Search Function

The backchannel app achieves the search function by using a search box on the top of homepage. Any user, including logged-in user and guest, can search the posts by users created the posts or posts' content.

The guest can only view the result of the search. While, the logged-in users can not only view the posts, but also vote, comment the corresponding posts according to their limits of authority.

If there is no post meets the requirements, the result will display no post. The user can back to the view list of all posts at any time.

==Voting System and Post Activity

Any logged-in user can vote for posts and replies. Logged-in user cannot vote for posts created by him/herself. No post can be voted by the same user more than once. Every post and comment has a count of votes.

The post with the largest count of votes will be identified as the "most active" post, and they are displayed at the top of list. The "Most active" is determined by the number of votes, as well as time elapsed.

We make up a weighted metric based on the count of votes and the passed time：
WEIGHT = the count of votes – days since it created

Posts are ordered in the list by the weight. When the weight decays to 0, the post will be deleted, that means hidden from the display.

==Tables

Using the db:migrate functionality of rails, we create five database tables.

Table Users:

The table records the information of the user account, including user id, user name, password and authenticity. The attribute of authenticity represent the user is an ordinary user or an administrator. If the authenticity equals 1, this is an administrator, while 0 means ordinary user. When admin create a new admin, the authenticity will be set as 1 automatically, while the other creation of a new user, it will be set as 0.

Table Posts:

The table contains the information related with the posts. The attributes id, title, content contain the basic information of the posts. The attributes postUserID and posterName record the information of user who creates this post. The attribute votes means the count of votes for this post. The created_at attribute is the date when the post is created. Based on attribute votes and created_at, we can get the weight for each post, which is used to order the display of posts.

Table Comments:

The table is used to record the information according to each comment. The attribute body contains the content of comment. The attribute commenterID record the userID who create this comment. The attribute post_id record the post that this comment is commenting to. The attribute commentVotes means the count of votes for the comment. The attribute created_at record the date time the comment is created.

Table Votes:

The table records the relationship among each post, its votes and the user who vote the post.

Each time a logged-in user is going to vote a post, we should first refer to the Posts table, if the user is not the owner of the post, then we refer to the Votes table. Selecting all the users who have voted this post by refer to the postID, if the user is inside the selection of users, which means the user has already voted this post, then return an announcement of error. Otherwise, we add the postID and userID to the table.

Table Comment_votes:

The table records the relationship among each comment, its votes, its post and the user who vote the comment.

Each time a logged-in user is going to vote a comment, we should first refer to the Comments table. If the user is not the owner of the comment, then we refer to the Comment_votes table. Selecting all the users who have voted this comment by refer to the postID and commentID, if the user is inside the selection of users, which means the user has already voted this comment, then return an announcement of error. Otherwise, we add the commentID, postID and userID to the table.

The above two tables allow us to prevent the same user from voting the same posts or comments more than once.

==Admin Account

The original administrator account is “admin”, with password “admin”. This account cannot be deleted by any other users.