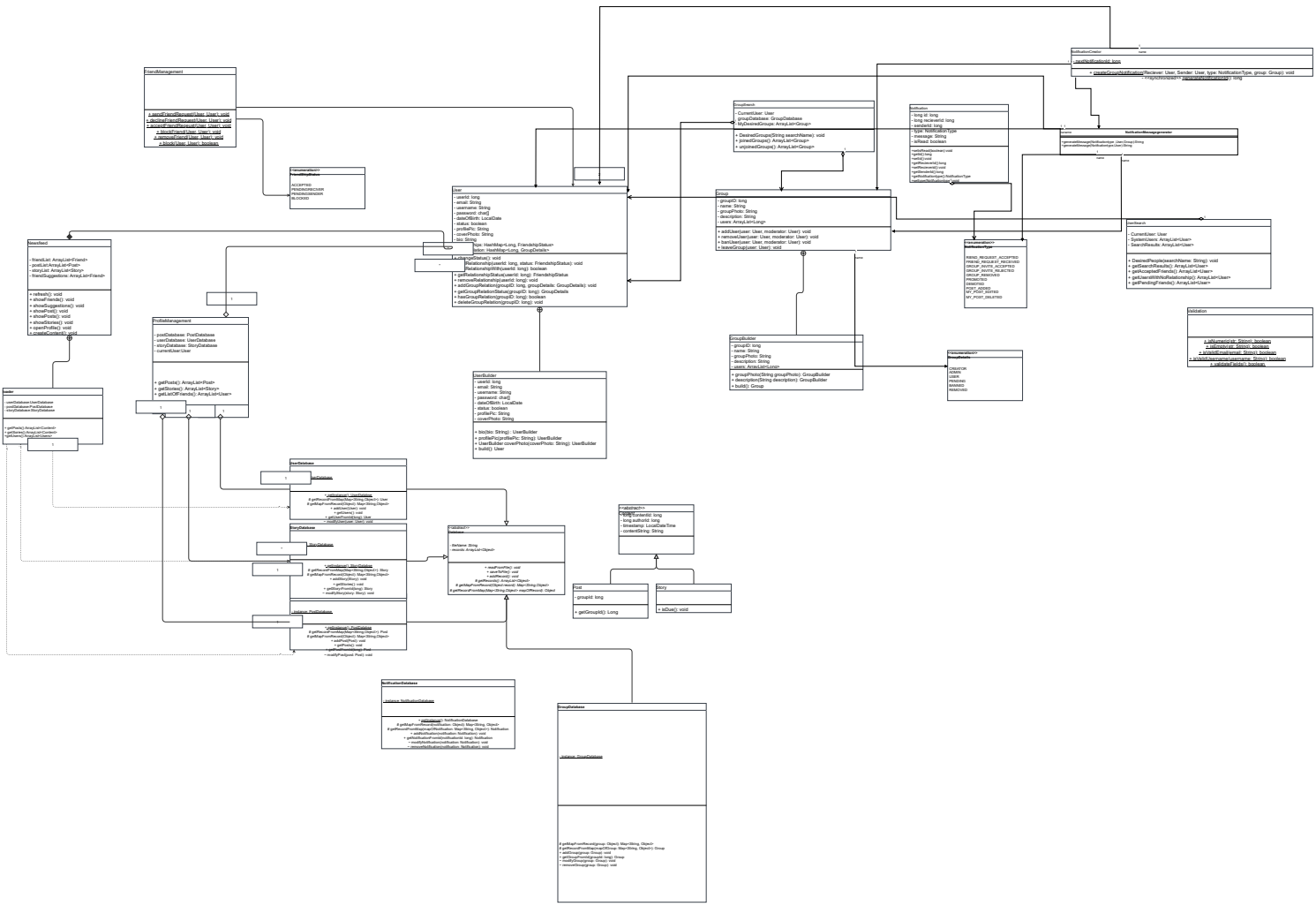
FriendManagement
+ sendFriendRequest(User, User): void
+ acceptFriendRequest(User, User): void
+ recendFriendRequest(User, User): void
+ blockFriend(User, User): void
+ removeFriend(User, User): void
+ isUser(User, User): boolean
<enumeration>>
FriendRequestStatus
ACCEPTED
PENDINGRECEIVER
PENDINGSENDER
BLOCKED
User
- userId: long
- email: String
- username: String
- password: char[]
- dateOfBirth: LocalDate
- status: boolean
- profilePic: String
- coverPhoto: String
- bio: String
- friendships: HashMap<Long, FriendshipStatus>
- relationship: HashMap<Long, GroupDetails>
+ getFriendshipStatus(): void
+ addRelationship(userId: long, status: FriendshipStatus): void
+ isRelationshipWith(userId: long): boolean
+ getRelationshipStatus(userId: long): FriendshipStatus
+ removeRelationship(userId: long): void
+ addGroupRelation(groupID: long, groupDetails: GroupDetails): void
+ getGroupRelationDetails(groupID: long): GroupDetails
+ isInGroupRelation(groupID: long): boolean
+ deleteGroupRelation(groupID: long): void
NewsFeed
- friendList: ArrayList<Friend>
- postList:ArrayList<Post>
- storyList: ArrayList<Story>
- friendSuggestions: ArrayList<Friend>
+ refresh(): void
+ showFriends(): void
+ showSuggestions(): void
+ showPost(): void
+ showPosts(): void
+ showStories(): void
+ openProfile(): void
+ createPost(): void
ProfileManagement
- postDatabase: PostDatabase
- userDatabase: UserDatabase
- storyDatabase: StoryDatabase
- currentUserUser
+ getPosts(): ArrayList<Post>
+ getStories(): ArrayList<Story>
+ getListOfFriends(): ArrayList<User>
Feeder
- userDatabase:UserDatabase
- postDatabase:PostDatabase
- storyDatabase:StoryDatabase
+ getPosts(): ArrayList<Content>
+ getStories(): ArrayList<Content>
+ getUsers(): ArrayList<User>
UserBuilder
- userId: long
- email: String
- username: String
- password: char[]
- dateOfBirth: LocalDate
- status: boolean
- profilePic: String
- coverPhoto: String
+ bio(bio: String): UserBuilder
+ profilePic(profilePic: String): UserBuilder
+ UserBuilder.coverPhoto(coverPhoto: String): UserBuilder
+ build(): User
UserDatabase
- instance: UserDatabase
+ getInstance(): UserDatabase
# getRecordFromRecordSet(rs: ResultSet): User
# getMapFromRecord(record: User): Map<String,Object>
+ addUser(User): void
+ getUser(): User
+ getUsersFromId(ids[]): User
+ removeUser(username: User): void
StoryDatabase
- instance: StoryDatabase
+ getInstance(): StoryDatabase
# getRecordFromMap(Map<String,Object>): Story
# getMapFromRecord(Story record): Map<String,Object>
+ addStory(Story): void
+ getStories(): void
+ getStoryFromId(id: long): Story
+ removeStory(story: Story): void
PostDatabase
- instance: PostDatabase
+ getInstance(): PostDatabase
# getRecordFromMap(Map<String,Object>): Post
# getMapFromRecord(Post record): Map<String,Object>
+ addPost(Post): void
+ getPosts(): void
+ getPostsFromId(id: long): Post
+ removePost(post: Post): void
<interface>>
IDatabase
- tableName: String
- records: ArrayList<Object>
+ readPost(id): void
+ saveToFile(): void
+ getTable(): void
# getRecord(): ArrayList<Object>
# getMapFromRecord(Object record): Map<String,Object>
# getRecordFromMap(Map<String,Object> mapOfRecord): Object
<abstract>>
Content
- contentId: long
- long authorId: long
- creationDateTime: LocalDateTime
- contentString: String
Post
- groupId: long
+ getGroupId(): Long
Story
+ isDue(): void
NotificationDatabase
- instance: NotificationDatabase
+ getInstance(): NotificationDatabase
# getMapFromRecord(Object record): Map<String, Object>
# getRecordFromMap(mapOfNotification: Map<String, Object>): Notification
+ addNotification(notification:Notification): void
+ getNotificationFromId(notificationId: long): Notification
+ modifyNotification(notification: Notification): void
+ removeNotification(notification: Notification): void
GroupDatabase
- instance: GroupDatabase
getMapFromRecord(group: Object): Map<String, Object>
getRecordFromMap(mapOfGroup: Map<String, Object>): Group
+ addGroup(group: Group): void
+ getGroupFromId(groupId: long): Group
+ modifyGroup(group: Group): void
+ removeGroup(group: Group): void
GroupSearch
- CurrentUser: User
- groupDatabase: GroupDatabase
- MyJoinedGroups: ArrayList<Group>
+ DesiredGroups(String searchName): void
+ joinedGroups(): ArrayList<Group>
+ unjoinedGroups(): ArrayList<Group>
Notification
- long id: long
- long recieverId: long
- senderId: long
- NotificationType type
- message: String
- isRead: boolean
+ setIsRead(boolean):void
+ getId()
+ isRd()
+ getRecieverId()
+ getSenderId()
+ getMessage()
+ getNotificationType()
+ setNotificationType()
Group
- groupID: long
- name: String
- groupPhoto: String
- description: String
- users: ArrayList<Long>
+ addUser(user: User, moderator: User): void
+ removeUser(user: User, moderator: User): void
+ banUser(user: User, moderator: User): void
+ leaveGroup(user: User): void
GroupBuilder
- groupID: long
- name: String
- groupPhoto: String
- description: String
- users: ArrayList<Long>
+ groupPhoto(String groupPhoto): GroupBuilder
+ description(String description): GroupBuilder
+ build(): Group
<enumeration>>
GroupRole
CREATOR
ADMIN
USER
PENDING
BANNED
RMOVED
<enumeration>>
NotificationType
FRIEND_REQUEST_ACCEPTED
FRIEND_REQUEST_RECEIVED
GROUP_INVITE_RECEIVED
GROUP_INVITE_REJECTED
GROUP_REMOVED
GROUP_BAN
PROMOTED
DEMOTED
POST_ADDED
MY_POST_REPORTED
MY_POST_DELETED
NotificationMessageGenerator
+ generateMessage(NotificationType: User,Group):String
+ generateMessage(NotificationType:User):String
NotificationCreator
- newNotificationId: long
+ createGroupNotification(Reciever: User, Sender User, type: NotificationType, group: Group): void
- ++nlyichonizeid++ newNotificationId(): long
SearchBarch
- CurrentUser: User
- SystemUsers: ArrayList<User>
- SearchResults: ArrayList<User>
+ DesiredPeople(searchName: String): void
+ getSearchResults(): ArrayList<User>
+ getAcceptedFriends(): ArrayList<User>
+ getUsersWithNoRelationship(): ArrayList<User>
+ getPendingFriends(): ArrayList<User>
Validation
+ isNumeric(str: String): boolean
+ isEmpty(str: String): boolean
+ isValidEmail(email: String): boolean
+ isValidUsernameUsername: String): boolean
+ validateFields(): boolean