TO DO	DOING	DONE
	Create schemas, routes and controllers *In progress*	**Create schemas** *Completed*
	Documentation *In progress*	**Design state transitions** *Completed*
	Add comments to all controllers and routes *In progress*	**List out some endpoints** *Completed*
	Define endpoints *In progress*	**Prepare server for coding and push to git** *Completed*
		Get a name for product *Completed*
		Create code for analysis *Completed*
		Gather the necessary status codes *Completed*
		Upload images and store in DB *Completed*

Reminders

1. Always push to github frequently.
2. Flow the manuscript, if you make changes, update it.
3. Always have usability in mind.
4. Deadline is 07/01/2022.
5. As lightweight as possible.
6. Create semantic profile (talk about how the data is extracted).
7. Find a way to separate access between user and admin.

Analytics(dev)

1. Amount of users created.
2. Amount of messages created.
3. Top 5 users with highest amount of messages.
4. Give a summary analysis.
5. Periods / days of frequent usage.

Endpoints

1. Create users.
2. View total users (analytics).
3. List of all available users to be added as friends.
4. View total messages (analytics).
5. View all friends.
6. View friends list/requests. Also sent requests.
7. View messages.

Features

1. Users should be able to add friends.
2. Users can see read and unread messages, messages are ordered by time.
3. Users are able to send messages.
4. Third-party developers can view analytics.
5. End-to-End encryption.

Analytics(user)

1. Total messages (separate between sent and received).
2. Total number of friends.
3. Filter read/unread messages for users.

GOAL	The goal of the API is to incorporate a lightweightchat server that can be consumed by frontend applications.

Functional Requierment	1. Security: No unauthorised usage. 2. Privacy: Message will not be received by wrong user.

Non-Functional Requierment	1. Performance. 2. Usabiity. 3. As steep as possible learning curve.

Friends

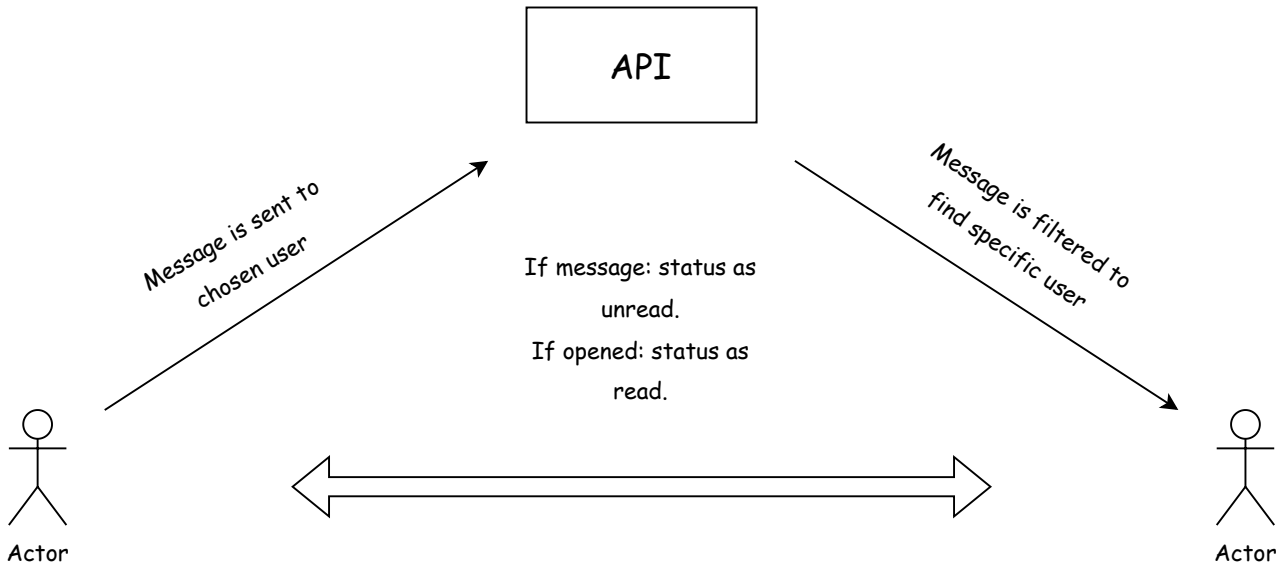
1. Be able to send friend requests.
2. Be able to receive friend requests.
3. View friends list & request list.
4. Search for a specific friend.
5. Only your friend can message you.
6. Encrypt API key and decrypt upon usage.

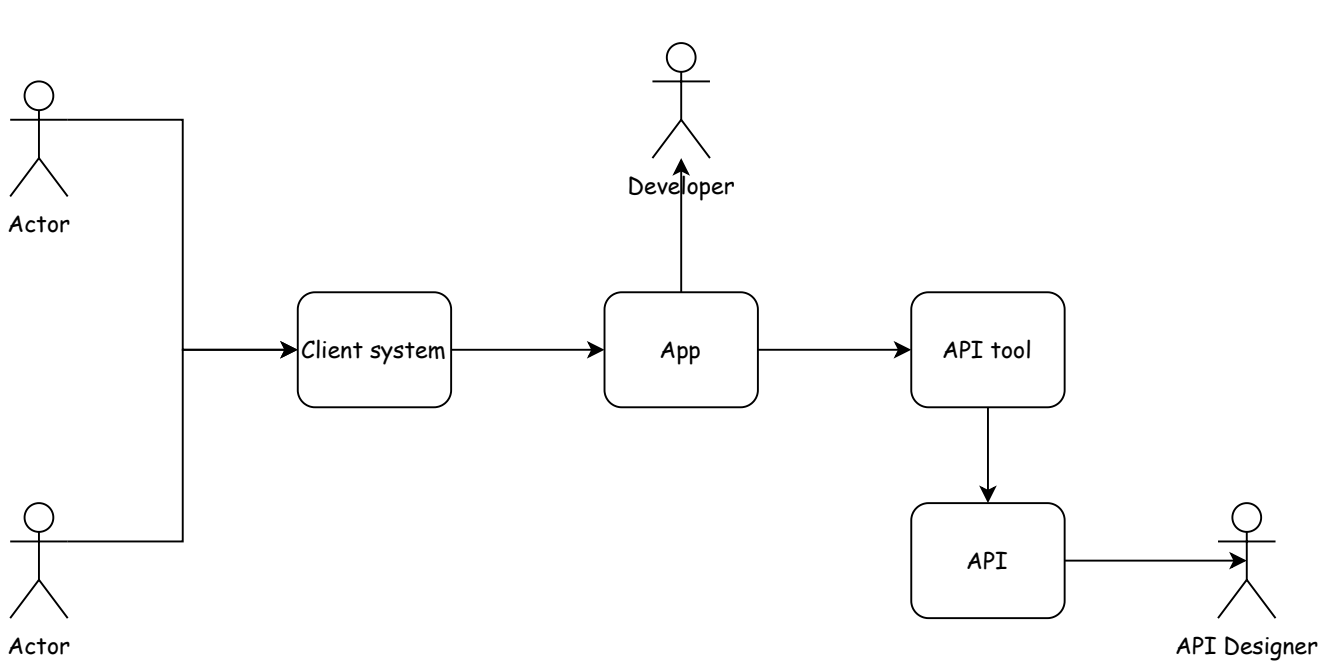
User

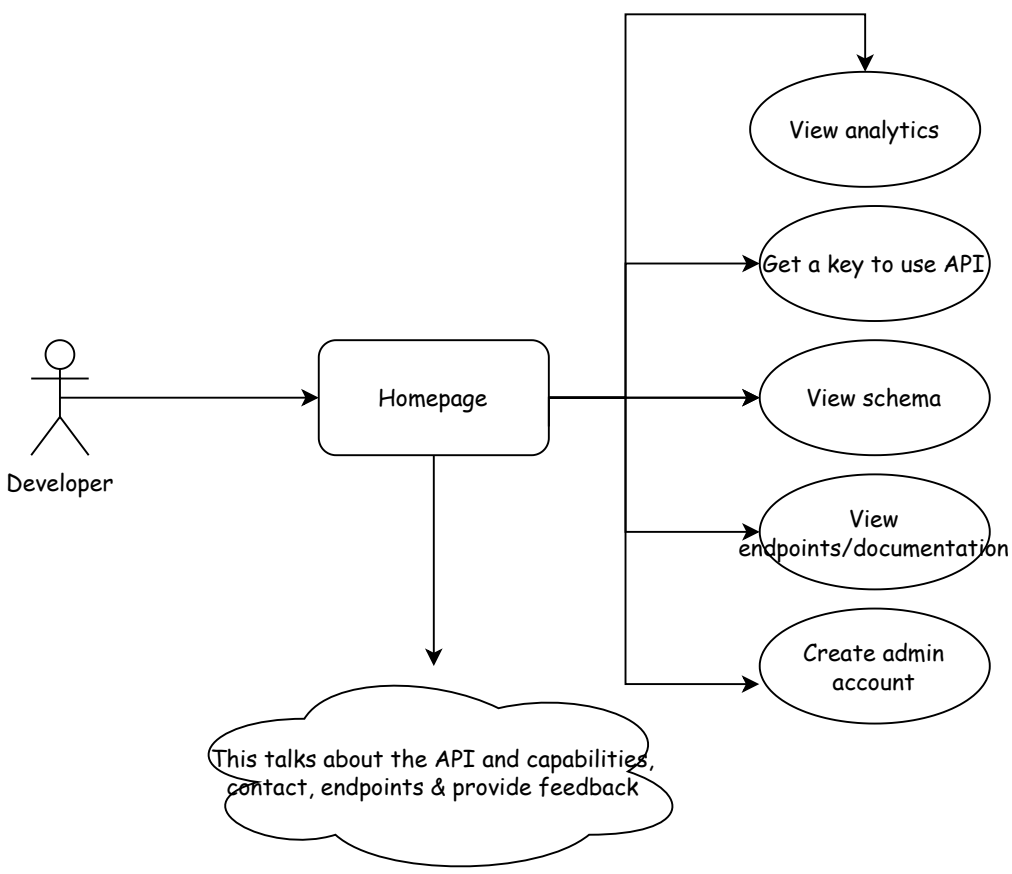
1. View all available users.
2. Search for a specific user.
3. Can modify profile.

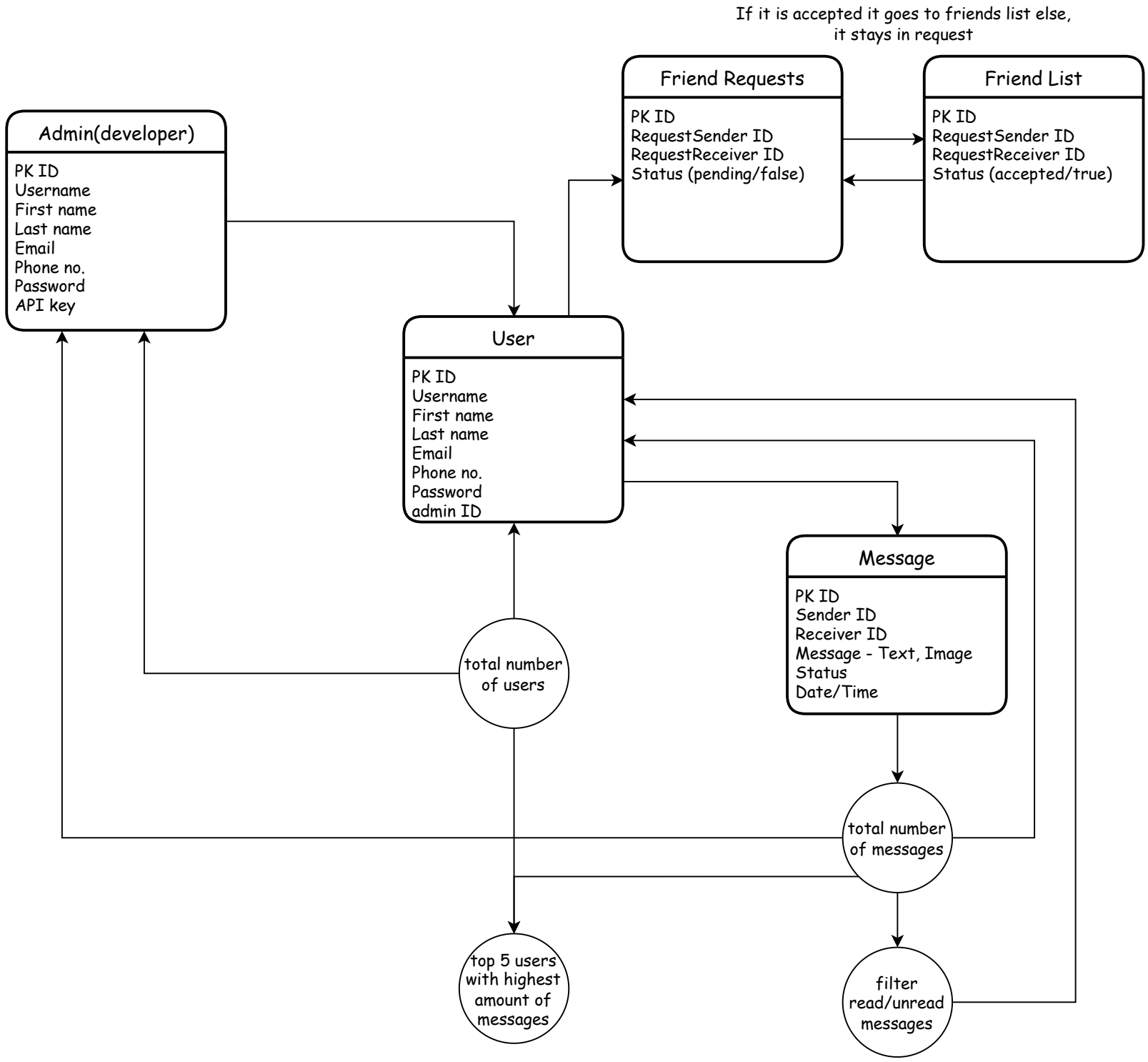
Messages

1. Create messages.
2. Filter messages.
3. Create messages with images.
4. Messages can automatically delete after 24 hours.


API
Message is sent to chosen user
Message is filtered to find specific user
If message: status as unread.
If opened: status as read.
Actor
Actor

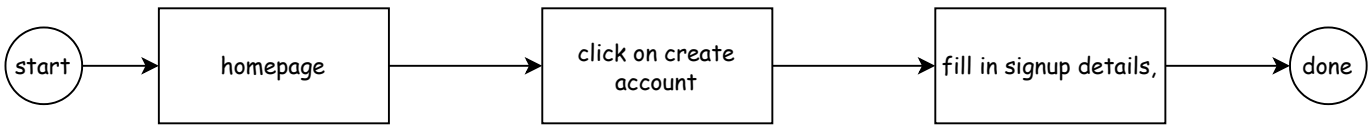

Actor
Developer
Client system
App
API tool
Actor
API
API Designer


View analytics
Get a key to use API
Developer
Homepage
View schema
View endpoints/documentation
Create admin account
This talks about the API and capabilities, contact, endpoints & provide feedback

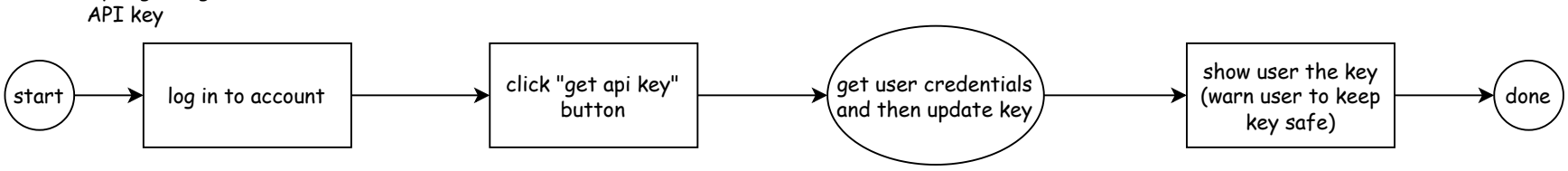

If it is accepted it goes to friends list else,
it stays in request
Friend Requests
PK ID
RequestSender ID
RequestReceiver ID
Status (pending/false)
Friend List
PK ID
RequestSender ID
RequestReceiver ID
Status (accepted/true)
Admin(developer)
PK ID
Username
First name
Last name
Email
Phone no.
Password
API key
User
PK ID
Username
First name
Last name
Email
Phone no.
Password
admin ID
Message
PK ID
Sender ID
Receiver ID
Message - Text, Image
Status
Date/Time
total number of users
total number of messages
top 5 users with highest amount of messages
filter read/unread messages

Homepage flow

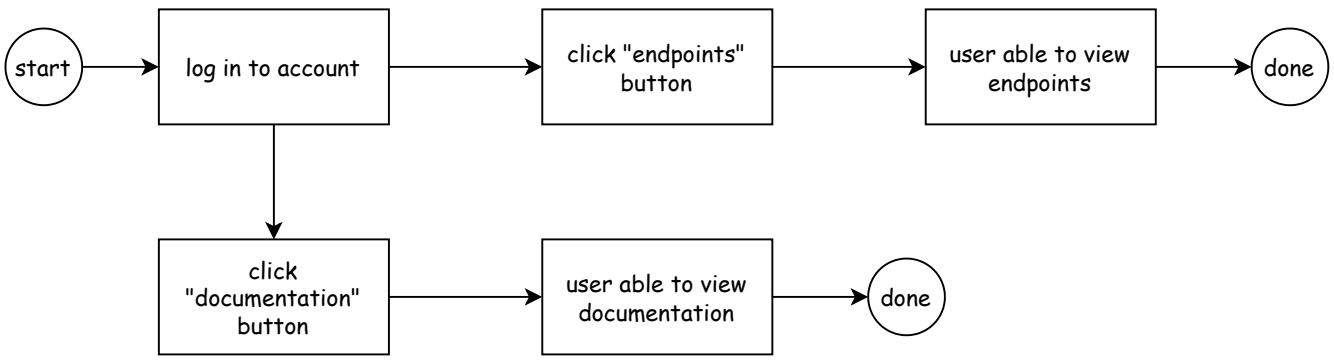
creating admin(developer) account



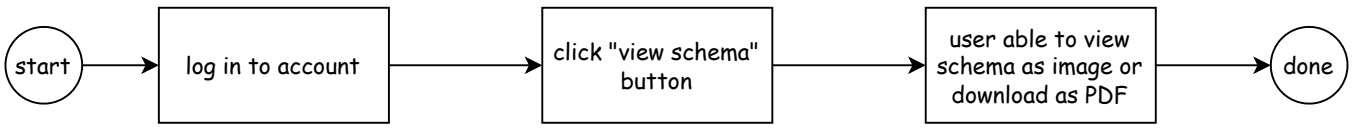
developer getting API key



developer viewing endpoints/documentation



developer viewing schema



developer viewing analytics