tautchat (team 5)

Taut tightens where Slack slacks

● ● ●

Alexander Showalter-Bucher

Steven Swanton
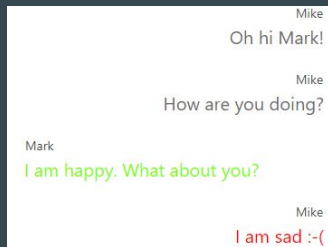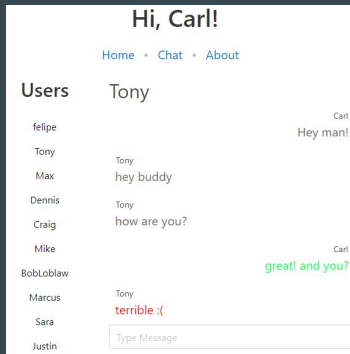
Dayton Wilson

Chad Woodrow

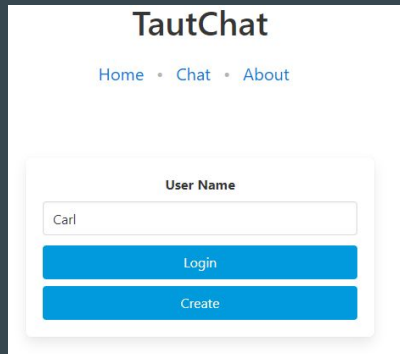# tautchat Highlights
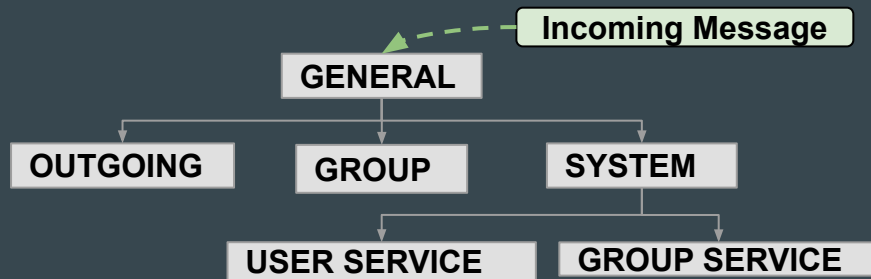
Chat with Clarity of Emotions
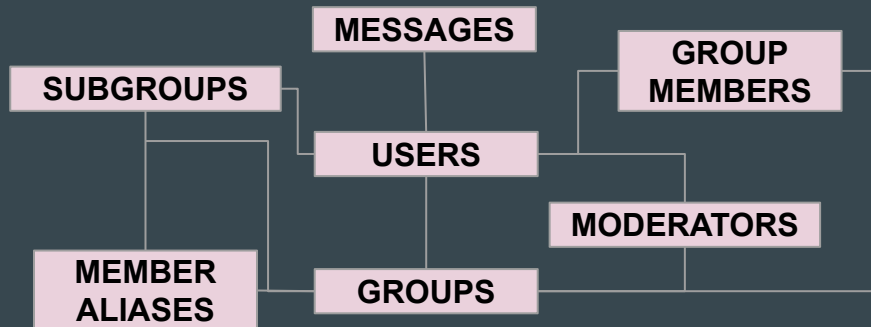
Sentiment Analysis of

- Group Chats
- User Chats

Mike
Oh hi Mark!

Mike
How are you doing?

Mark
I am happy. What about you?

Mike
I am sad :-(

Modular/Extendable Messaging Bus

Incoming Message

GENERAL

OUTGOING   **GROUP**   **SYSTEM**

USER SERVICE   **GROUP SERVICE**

Clean/Friendly Interface (ReactJS)

TautChat

Home · Chat · About

User Name

Carl

Login

Create

Hi, Carl!

Home · Chat · About

Users   Tony

felipe
Tony   Carl
Max   Tony   Hey man!
Dennis   hey buddy
Craig   Tony
Mike   how are you?
BobLoblaw   Carl
Marcus   great! and you?
Sara   Tony
Justin   terrible :(

Type Message

Persistent of Data via AWS MYSQl Database

MESSAGES

SUBGROUPS   **GROUP MEMBERS**

USERS

MEMBER ALIASES   **MODERATORS**

GROUPS

UML Progression
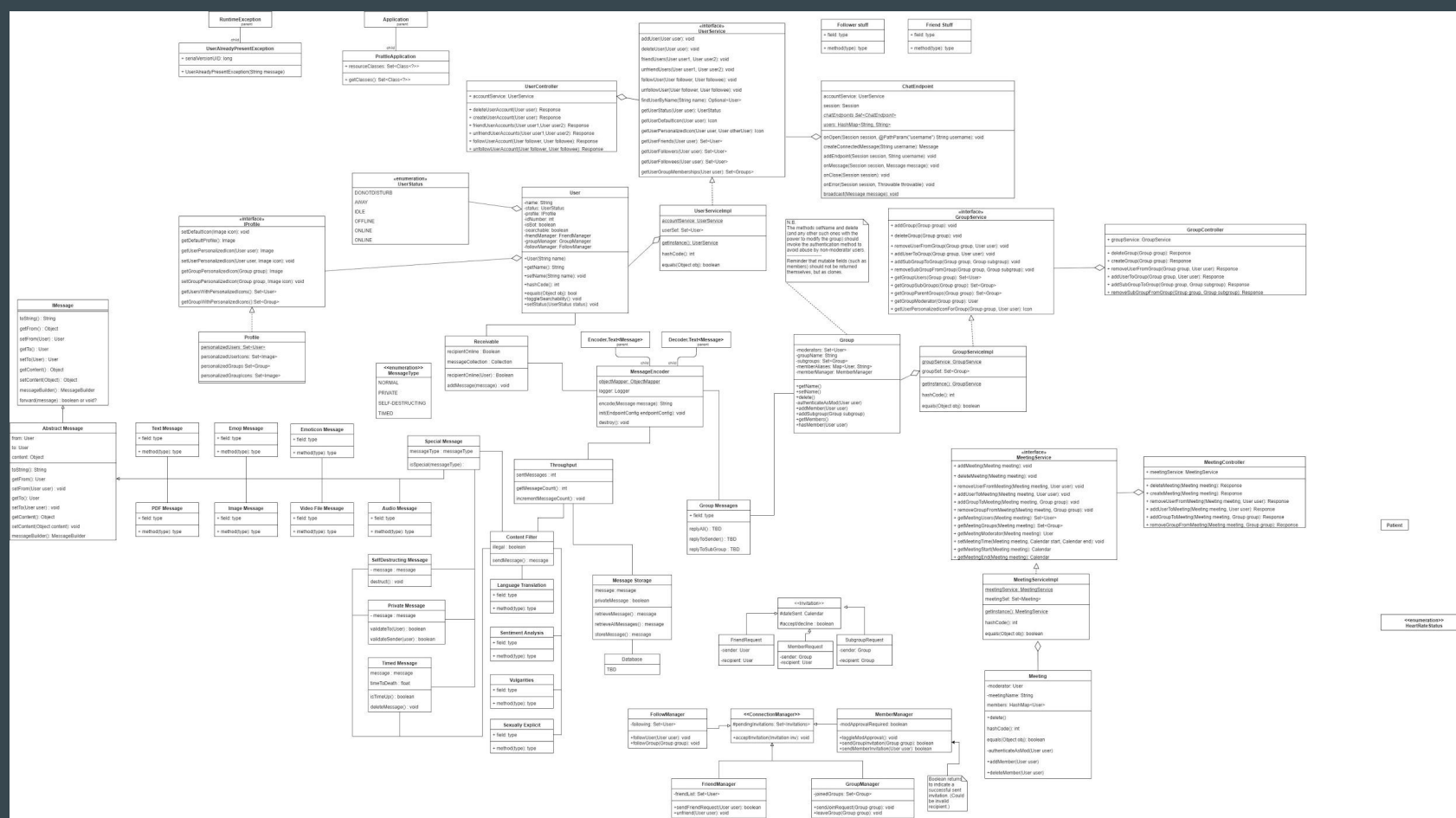
User Stories/Tasks : Initial

- Source Code tests
- Add Operational Requirements to SRS
- Team Review Source Code
- First Draft UML
- Install Tomcat
- Select Host Platform
- Users may send messages to other specific users
- Users may send messages to groups
- Users can access sent messages
- Users may join groups
- Users can have personalized icons
- User icons can be made specific to a certain group
- Users may id other users as friends
- Users can see other users with mutual friends
- Users can mark themselves in a Do Not Disturb state
- Users can mark themselves in an Away state
- Users will be marked with an Idle state
- Users will be marked with an Offline designation
- Users will be marked with an Online designation
- Users may set themselves as searchable or not
- Users may follow other users 'posts' TBD
- Users may follow groups' 'posts' TBD
- Users may recall or delete sent messages if they remain unread
- Groups have a moderator user associated with them
- Moderator Users may remove users from groups they moderate
- Moderators can approve/deny users joining the group they are moderating
- Moderators may delete groups they moderate
- Moderators may approve/deny other groups from joining the group the moderator moderates
- Users may invite other users to join a Group

- Users may invite other Groups to join a group they are a member of
- Groups allow members to message the entire group
- All messages will be stored in perpetuity
- Messages will queue when recipient-user is offline
- Persistence
- Two users in the same group may message each other individually
- Groups allow member users to message a subset of Group users
- Notification sent to Slack if service crashes
- Service will send a heartbeat signal for monitoring purposes
- Count of active users will be tracked
- Count of messages sent will be tracked
- Service will track throughput rates of messages
- Service will track resource utilization
- Tracked metrics will be available to external query
- Users have a unique login to access their account
- Login may be authorized through external service
- Groups have the option to require a password to access
- Service tiers determine how long user messages are available to the user
- Messages will be end-to-end encrypted
- Encryption will have a paid aspect
- Messages will have the ability to set for auto-deletion after a set time/event
- Multiple language character sets are supported
- Non-Latin character sets are supported
- Messages may contain emojis
- Queued messages shall be received by recipient upon their subsequent login
- Messages may contain emoticons
- Messages may contain audio files
- Messages may contain video files

- Messages may contain image files
- Messages may contain pdf files
- Service will allow access to real-time communication of subpoenaed users
- Login/Logoff of subpoenaed users will be tracked
- Decryption of messages will be done client-side
- Service will offer a parental controls/content filter
- Users may choose to filter objectionable content
- Filter will function server-side
- Filter will provide multiple methods of filtering
- Users may send 'private' messages
- Private messages cannot be forwarded to other users
- Private messages may not be copied
- Private messages may be encrypted during transport
- Private messages may be encrypted in storage
- Hashtags will allow messages to be categorized
- Users may search for messages containing specific hashtags
- Users may see current/active hashtags
- Users may schedule meetings
- Users may schedule events
- Meetings/Events may request RSVPs from invitees
- Users can register a health tracker to their user account
- Users may register a health tracker to a group
- Health tracker will transmit temporal heartbeat rate to service
- Health tracker will transmit GPS location
- Users may be notified if there is sudden change in heart rate data
- User may be notified of location of health tracker
- Users may forward messages from another user to a third user
- Message sentiment polarity will be displayed to group

- Authors of 'forwarded' messages may track where/to whom their message is forwarded to
- GUI integration
- GUI should be configured to accommodate these changes
- Should be able to add/login as user (no authentication)
- Messages will be analyzed for sentiment polarity
- Users should be able to associate with other users as friends
- Front end messaging system to support communication with the backend
- Outline slides for presentation
- Everyone should update their parts of the UML to reflect our work
- Resolve timeout issue for testing so code can be pushed to master
- Remove add subgroup/supergroup from Group builder
- Messages can auto-translate text
- Message sentiment polarity will be displayed to user
- Users may enable/disable sentiment analysis
- Interface should exist for sending/receiving direct chat to users
- Interface should exist to search for users to message them
- Chat interface should be able to display sentiment analyzed messages
- Interface should exist to add friends
- Interface should exist to join groups
- Interface should exist to add groups
- Interface should exist to moderate groups
- Interface for text translation
- Develop slides for presentation
- Generate animated diagram of how the message bus works
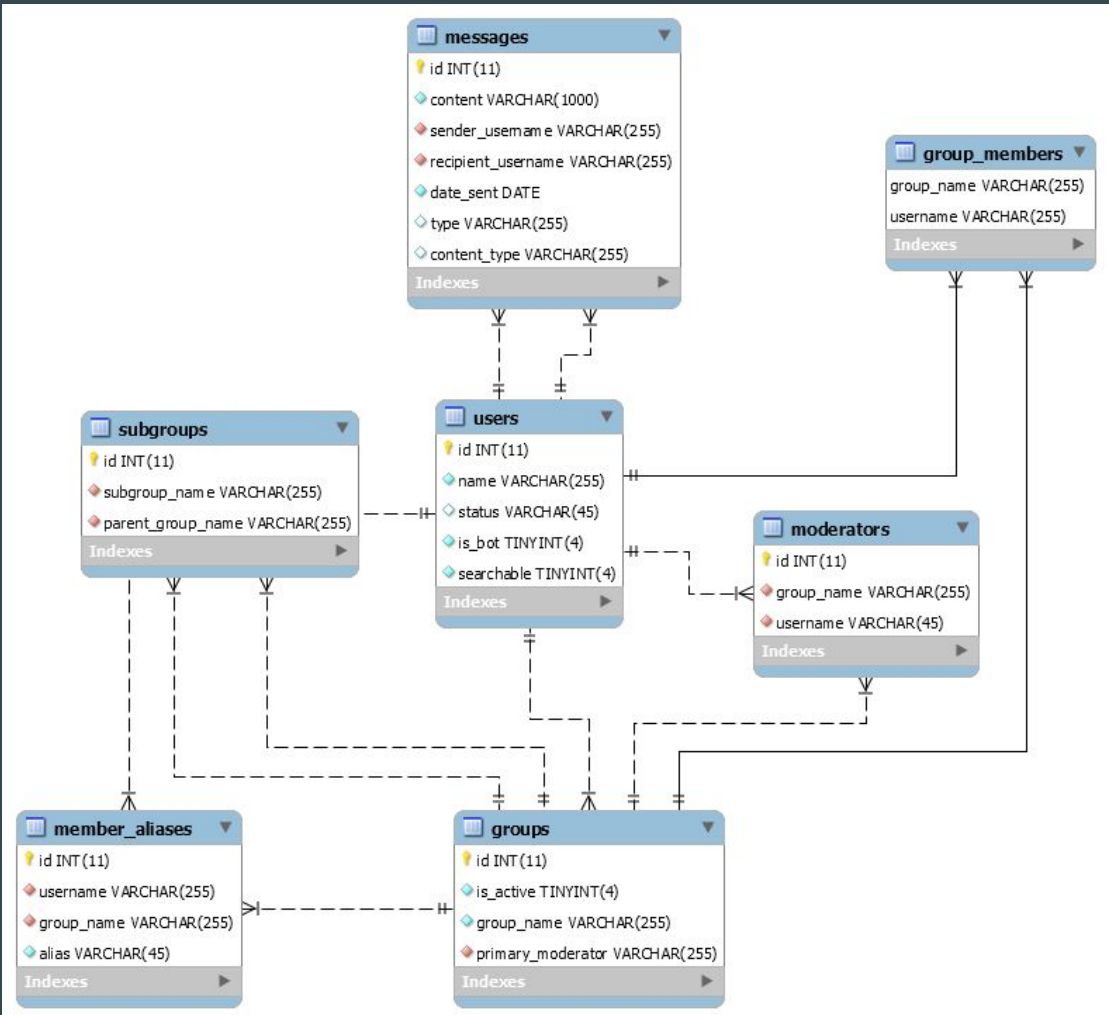- Connect text translation

User Stories/Tasks : Progress

- Source Code tests
- Add Operational Requirements to SRS
- Team Review Source Code
- First Draft UML
- Install Tomcat
- Select Host Platform
- Users may send messages to other specific users
- Users may send messages to groups
- Users can access sent messages
- Users may join groups
- Users can have personalized icons
- User icons can be made specific to a certain group
- Users may id other users as friends
- Users can see other users with mutual friends
- Users can mark themselves in a Do Not Disturb state
- Users can mark themselves in an Away state
- Users will be marked with an Idle state
- Users will be marked with an Offline designation
- Users will be marked with an Online designation
- Users may set themselves as searchable or not
- Users may follow other users 'posts' TBD
- Users may follow groups' 'posts' TBD
- Users may recall or delete sent messages if they remain unread
- Groups have a moderator user associated with them
- Moderator Users may remove users from groups they moderate
- Moderators can approve/deny users joining the group they are moderating
- Moderators may delete groups they moderate
- Moderators may approve/deny other groups from joining the group the moderator moderates
- Users may invite other users to join a Group

- Users may invite other Groups to join a group they are a member of
- Groups allow members to message the entire group
- All messages will be stored in perpetuity
- Messages will queue when recipient-user is offline
- Persistence
- Two users in the same group may message each other individually
- Groups allow member users to message a subset of Group users
- Notification sent to Slack if service crashes
- Service will send a heartbeat signal for monitoring purposes
- Count of active users will be tracked
- Count of messages sent will be tracked
- Service will track throughput rates of messages
- Service will track resource utilization
- Tracked metrics will be available to external query
- Users have a unique login to access their account
- Login may be authorized through external service
- Groups have the option to require a password to access
- Service tiers determine how long user messages are available to the user
- Messages will be end-to-end encrypted
- Encryption will have a paid aspect
- Messages will have the ability to set for auto-deletion after a set time/event
- Multiple language character sets are supported
- Non-Latin character sets are supported
- Messages may contain emojis
- Queued messages shall be received by recipient upon their subsequent login
- Messages may contain emoticons
- Messages may contain audio files
- Messages may contain video files

- Messages may contain image files
- Messages may contain pdf files
- Service will allow access to real-time communication of subpoenaed users
- Login/Logoff of subpoenaed users will be tracked
- Decryption of messages will be done client-side
- Service will offer a parental controls/content filter
- Users may choose to filter objectionable content
- Filter will function server-side
- Filter will provide multiple methods of filtering
- Users may send 'private' messages
- Private messages cannot be forwarded to other users
- Private messages may not be copied
- Private messages may be encrypted during transport
- Private messages may be encrypted in storage
- Hashtags will allow messages to be categorized
- Users may search for messages containing specific hashtags
- Users may see current/active hashtags
- Users may schedule meetings
- Users may schedule events
- Meetings/Events may request RSVPs from invitees
- Users can register a health tracker to their user account
- Users may register a health tracker to a group
- Health tracker will transmit temporal heartbeat rate to service
- Health tracker will transmit GPS location
- Users may be notified if there is sudden change in heart rate data
- User may be notified of location of health tracker
- Users may forward messages from another user to a third user
- Message sentiment polarity will be displayed to group

- Authors of 'forwarded' messages may track where/to whom their message is forwarded to
- GUI integration
- GUI should be configured to accommodate these changes
- Should be able to add/login as user (no authentication)
- Messages will be analyzed for sentiment polarity
- Users should be able to associate with other users as friends
- Front end messaging system to support communication with the backend
- Outline slides for presentation
- Everyone should update their parts of the UML to reflect our work
- Resolve timeout issue for testing so code can be pushed to master
- Remove add subgroup/supergroup from Group builder
- Messages can auto-translate text
- Message sentiment polarity will be displayed to user
- Users may enable/disable sentiment analysis
- Interface should exist for sending/receiving direct chat to users
- Interface should exist to search for users to message them
- Chat interface should be able to display sentiment analyzed messages
- Interface should exist to add friends
- Interface should exist to join groups
- Interface should exist to add groups
- Interface should exist to moderate groups
- Interface for text translation
- Develop slides for presentation
- Generate animated diagram of how the message bus works
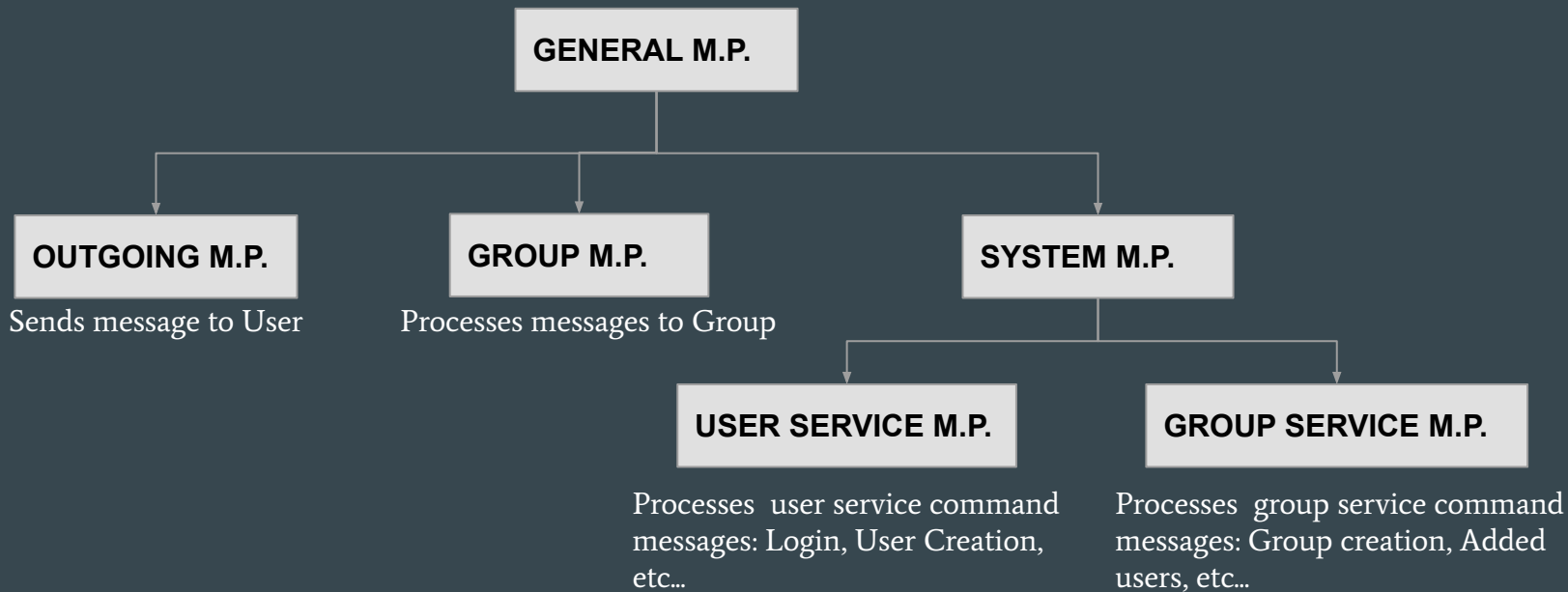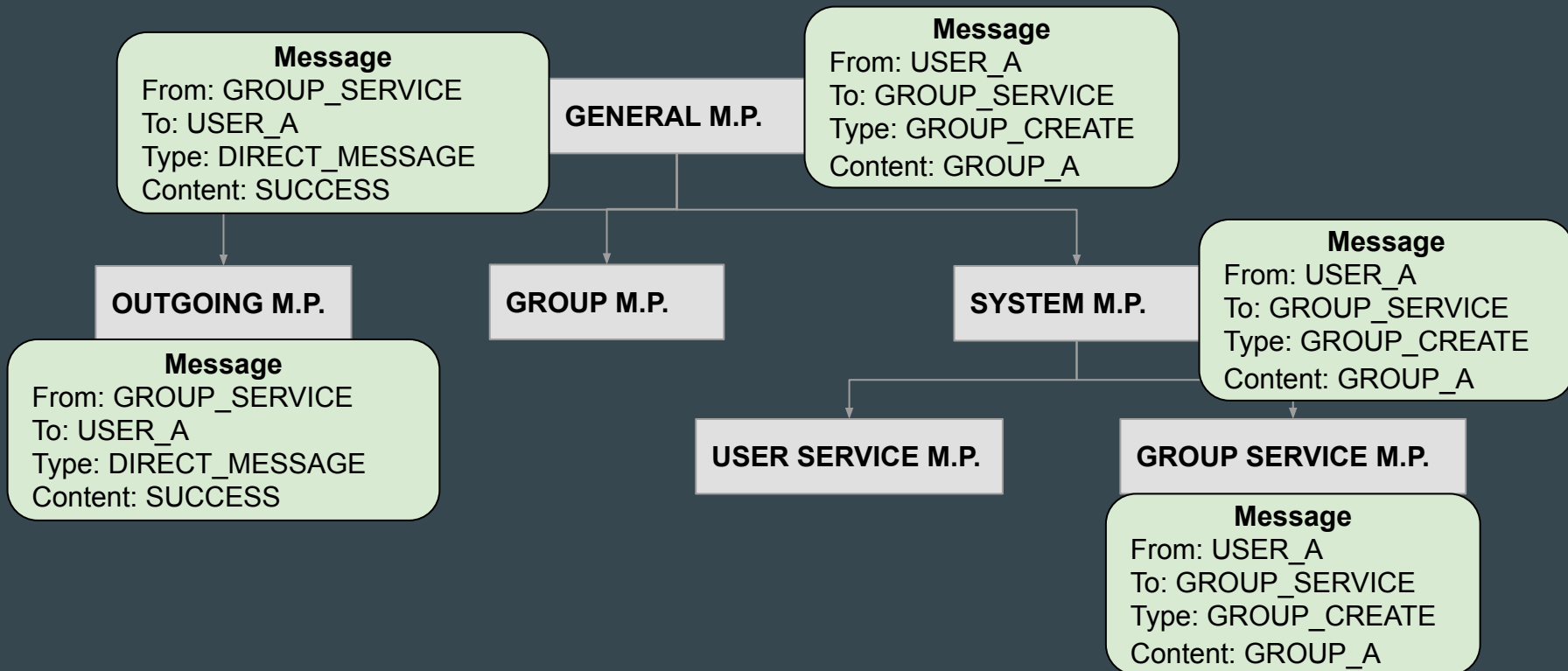- Connect text translation

UML Progression

UML Progression

Database Schema

Messaging Processing Framework

Example: Group Creation Message

Message
From: GROUP_SERVICE
To: USER_A
Type: DIRECT_MESSAGE
Content: SUCCESS

GENERAL M.P.

Message
From: USER_A
To: GROUP_SERVICE
Type: GROUP_CREATE
Content: GROUP_A

OUTGOING M.P.

GROUP M.P.

SYSTEM M.P.

Message
From: USER_A
To: GROUP_SERVICE
Type: GROUP_CREATE
Content: GROUP_A

Message
From: GROUP_SERVICE
To: USER_A
Type: DIRECT_MESSAGE
Content: SUCCESS

USER SERVICE M.P.

GROUP SERVICE M.P.

Message
From: USER_A
To: GROUP_SERVICE
Type: GROUP_CREATE
Content: GROUP_A

Evidence of the Quality of Code

- Functionality:
  - Backlog of core backend functionality near completion
  - Integration of backend functionality with frontend still in progress
- Usability
  - Website accessible : www.tautchat.com
  - Intuitive interface build via ReactJS
- Maintainability (SonarQube Rating: A)
  - 0% duplicated lines/technical debt/No Smells
  - Followed "Law of Demeter" where possible.
  - Utilized design patterns heavily (Builder, Factory, Singleton, etc...)
- Efficiency
  - Only tested internally/ no estimations of performance at scale (outside current scope of project)
- Dependability/ Reliability /Resiliency
  - Solid Unit Testing of Java Backend (Line: 96.9% Condition: 54.4%*)
  - No explicit unit testing of javascript (outside current scope of project)

*Driven by potential SQL error conditions

tautchat Team Background

Dayton

Steve

Chad

Alex

- Unique Background Experience
  - 1 MS Mech. Eng. student
  - 2 ALIGN Program students
  - 1 Software Dev. experience (small scale project without proper SDLC practices)
- Unique Logistical Challenges
  - 3 working professionals
  - 1 Long-distance Commuter (Rhode Island)

Teaming Process

- Communication Logistics
  - General Comms: Slack
  - Scrums: Standup_Alice / slack communication
- Team Meeting Logistics
  - ~ 3x meetings a week (Sprint Review, Tuesdays, 1 additional day)
  - Remote Meetings: Microsoft Teams
- Sprint/Project Management: Jira
- Code Collaboration: NEU CSS Github
  - Branch Management: (Master ←→ Sprint ←→ Personal)

Sprint Retrospective (1/2)

| Sprint	Roadblocks	Team Accomplishments
1	- Scope - Meeting consistently	- Clarified many SRS requirements - Organized as functional/non-functional - Determined relative priority
2	Scope of sprint expectations	- Created UML for backend functionality - Hosted prattle on AWS - Added operational requirements
3	- Team not being on same page for goals - Focus drifting to less pertinent requirements - Weak scrumming	- Having discussions about individual issues - Better pre-planning for future sprints - Re-centered focus on implementing base functionalities for messaging and groups

Sprint Retrospective (2/2)

Sprint	Roadblocks	Team Accomplishments
3	● Differing levels of experience => Division of responsibility opaque ● Difficulty in merging individual work branches	● Prototyped GUI ● Initial implementation of message routing
4	● Lagging behind on deliverables/demo ● Pushing to master	● Database persistence ● Improved message routing ● Group functionality
5	● Syncing changes up to server ● GUI implementation	● Private messaging & sentiment analysis reflected in the GUI ● (ongoing…)

Demo

- ReactJS based User Interface
  - Utilized the following plugins:
    - Bulma (CSS), WebSocket, React Router
  - The following hosts were used for development
    - Local Development: http://localhost:3000/prattle
    - Group Development/Deployment: http://localhost:8080/prattle (Tomcat7)
    - Public Use: http://www.tautchat.com
  - Front End created to show off the following back end features:
    - Persistent Users
    - Private Messaging
    - Group Messaging
    - Group Creation
    - Sentiment Analysis
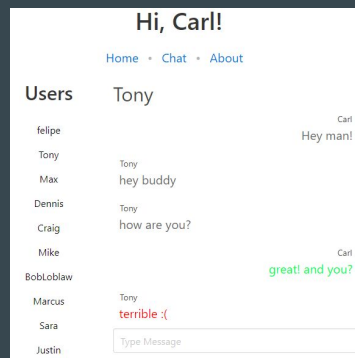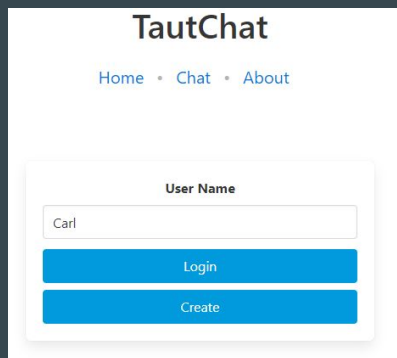
Next Steps

- Continue Front End Implementation (See below)
- Assess front end for scalability upon deployment

Feature	Back End	Front End
User Services (Create users, login)	Complete	Complete
Messaging (Private Messaging, Group Messaging)	Complete	Complete
Group Services (Add Groups, Set Mods, Add Users, Delete Users, Subgroups, etc.)	Complete	In Progress
Sentiment Analysis	Complete	*Complete
Friending Services (Add/delete friends, search users and add)	Complete	Not Started
Persistent Users	Complete	Complete
Persistent Messages	Complete	Not Started
Persistent Groups	Complete	Not Started

Fin
...

SonarQube Report