USE CASES

Use Case Identification and History	
Use Case ID:	1
Use Case Name:	Creating new user
End Objective:	To create new user
User/Actor:	Any user
Trigger:	User
Frequency of Use:	For every new user

Preconditions
Prattle is running, and chatter is connected to it

Basic Flow		
Step	**User Actions**	**System Actions**
1	User enters command CREATE_USER <username> <password>	System create the user in database, password is encrypted and sends a message saying user created.

Exception Flow	
1	If username already exists, user gets message saying, “Username already exists” .

Post conditions
1. User gets created in the database

Business Rules
1. Username should always be unique

Use Case Identification and History	
Use Case ID:	2
Use Case Name:	Login user
End Objective:	To login a user
User/Actor:	Any user
Trigger:	User
Frequency of Use:	Very frequent

Preconditions
Prattle is running, and chatter is connected to it

Basic Flow		
Step	**User Actions**	**System Actions**
1	User enters command LOGIN_USER <username> <password>	System verifies whether username and password exist, if yes send login message, else invalid credentials message

Exception Flow	
1	If invalid credentials user get message saying, “Invalid Credentials” .

Post conditions
1. User logins

Business Rules
1. Username and password should match with one in database

Use Case Identification and History	
Use Case ID:	3
Use Case Name:	Updating user’s password
End Objective:	To update the user’s password
User/Actor:	Any user
Trigger:	User
Frequency of Use:	Less frequent

Preconditions
Prattle is running, and chatter is connected to it, and user is logged in

Basic Flow		
Step	**User Actions**	**System Actions**
1	User enters command UPDATE_PASSWORD <current password> <new password>	System updates the password and send success message, if password does not match it sends invalid password message

Exception Flow	
1	If current password is incorrect user get a message saying, “Entered password is wrong” .

Post conditions
1. User’s password gets updated

Business Rules
1. Provided current password should be same as the actual current password. 2. User should be logged in.

Use Case Identification and History	
Use Case ID:	4
Use Case Name:	Delete user
End Objective:	To delete user from the system
User/Actor:	Any user
Trigger:	User
Frequency of Use:	Less frequent

Preconditions
Prattle is running, and chatter is connected to it, and user is logged in

Basic Flow		
Step	**User Actions**	**System Actions**
1	User enters command DELETE_USER <current password>	System delete the user and send success message, if password does not match it sends invalid password message

Exception Flow	
1	If current password is incorrect user get a message saying, “Entered password is wrong” .

Post conditions
1. User gets deleted

Business Rules
1. Provided current password should be same as the actual current password. 2. User should be logged in.

Use Case Identification and History	
Use Case ID:	5
Use Case Name:	Add a new group
End Objective:	A new group added to the system
User/Actor:	Any user
Trigger:	User
Frequency of Use:	Less frequent

Preconditions
Prattle is running, and chatter is connected to it, and user is logged in

Basic Flow		
Step	**User Actions**	**System Actions**
1	User enters command CREATE_GROUP <group name>	System adds new group to the system and add creating user to that group. If group name already exists, group name already exist message is sent.

Exception Flow	
1	If group name already exists, message is sent saying , “Group name exists” .

Post conditions
1. New group added to the system.

Business Rules
1. Group name should always be unique 2. User should be logged in

Use Case Identification and History	
Use Case ID:	6
Use Case Name:	Join a group
End Objective:	The user gets added to the group
User/Actor:	Any user
Trigger:	User
Frequency of Use:	Less frequent

Preconditions
Prattle is running, and chatter is connected to it, and user is logged in

Basic Flow		
Step	**User Actions**	**System Actions**
1	User enters command JOIN_GROUP <group name>	System adds logged in user to the group. If group does not exist, user get error message.

Exception Flow	
1	If group does not exist, user gets message saying “This group does not exist” .

Post conditions
1. User gets added to the group.

Business Rules
1. Provided group name should be in the system. 2. User should be logged in

Use Case Identification and History	
Use Case ID:	7
Use Case Name:	Exit a group
End Objective:	The user get exits the group
User/Actor:	Any user
Trigger:	User
Frequency of Use:	Less frequent

Preconditions
Prattle is running, and chatter is connected to it, and user is logged in

Basic Flow		
Step	**User Actions**	**System Actions**
1	User enters command EXIT_GROUP <group name>	System remove logged in user from the group. If group does not exist, or user is not the part of the group, user get error message.

Exception Flow	
1	If group does not exist, user gets message saying, “This group does not exist” .
2	If user is not part of the group, user gets a message saying “Fail”

Post conditions
1. User exits the group.

Business Rules
1. Provided group name should be in the system. 2. User should be logged in

Use Case Identification and History	
Use Case ID:	8
Use Case Name:	Delete a group
End Objective:	Group get deleted from the system
User/Actor:	Any user
Trigger:	User
Frequency of Use:	Less frequent

Preconditions
Prattle is running, and chatter is connected to it, and user is logged in

Basic Flow		
Step	**User Actions**	**System Actions**
1	User enters command DELTE_GROUP <group name>	System delete the group, and send a success message.

Exception Flow	
1	If group does not exist, user gets message saying, “This group does not exist” .

Post conditions
1. User exits the group.

Business Rules
1. User should be logged in 2. Provided group name should be in the system.

Use Case Identification and History	
Use Case ID:	9
Use Case Name:	Send broadcast message
End Objective:	Broadcast message get sent
User/Actor:	Any user
Trigger:	User
Frequency of Use:	Frequent

Preconditions
Prattle is running, and chatter is connected to it, and user is logged in

Basic Flow		
Step	**User Actions**	**System Actions**
1	User enters command <message>	System sends a broadcast message to all active prattle connections.

Exception Flow	
1	If sending user is not logged in gets a message saying, “fail” .

Post conditions
1. Sender should be logged in

Business Rules
1. All active connection receives the broadcast message
2. Only logged in user can send a broadcast message

Use Case Identification and History	
Use Case ID:	10
Use Case Name:	Send a private message
End Objective:	Private message is sent to the user
User/Actor:	Any user
Trigger:	User
Frequency of Use:	Very Frequent

Preconditions
Prattle is running, and chatter is connected to it, and user is logged in

Basic Flow		
Step	**User Actions**	**System Actions**
1	User enters command PRIVATE <receiver name> <message>	System sends a private message to receiver and sends a unique message id back.

Exception Flow	
1	If sender user is not logged in, error message is displayed
2	If the receiver username does not exist in system, message saying "Username does not exist " is sent back

Post conditions
1. Private message is sent to the receiver and persisted in the database.

Business Rules
1. Sender user should be logged in 2. Receiver user name should exist in system.

Use Case Identification and History	
Use Case ID:	11
Use Case Name:	Send a group message
End Objective:	Group message is sent to all users of the group
User/Actor:	Any user
Trigger:	User
Frequency of Use:	Very Frequent

Preconditions
Prattle is running, and chatter is connected to it, and user is logged in

Basic Flow		
Step	**User Actions**	**System Actions**
1	User enters command GROUP <group name> <message>	System sends a group message to all members of the group and sends a unique message id back.

Exception Flow	
1	If sender user is not logged in, fail message is displayed
2	If the receiver group name does not exist in system, message saying “groupname does not exist ” is sent back
3	If sender user is not member of the group a fail message is displayed

Post conditions
1. Group message is sent to all members of the group and persisted in database

Business Rules
1. Sender user should be logged in 2. Group name should exist in system. 3. Sender user should be member of the group to which message is sent

Use Case Identification and History	
Use Case ID:	12
Use Case Name:	Send a MIME message
End Objective:	MIME message is sent to the user
User/Actor:	Any user
Trigger:	User
Frequency of Use:	Frequent

Preconditions
Prattle is running, and chatter is connected to it, and user is logged in

Basic Flow		
Step	**User Actions**	**System Actions**
1	User enters command MIME <receiver name> < file path>	System sends a private MIME message to receiver and sends a unique message id back.

Exception Flow	
1	If sender user is not logged in, error message is displayed
2	If the receiver username does not exist in system, message saying "Username does not exist " is sent back

Post conditions
1. MIME message is set to the receiver and persisted in the database.

Business Rules
1. Sender user should be logged in 2. Receiver user name should exist in system.

Use Case Identification and History	
Use Case ID:	13
Use Case Name:	Search a message
End Objective:	Message is searched for a user
User/Actor:	Any user
Trigger:	User
Frequency of Use:	Frequent

Preconditions
Prattle is running, and chatter is connected to it, and user is logged in

Basic Flow

Step	User Actions	System Actions
1	User enters command SEARCH <”Sender”/”receiver”> < username>	System searches for the messages of the user mentioned and send them back

Exception Flow	
1	If sender user is not logged in, error message is displayed
2	If the receiver username does not exist in system, message saying “Username does not exist ” is sent back

Post conditions
1. MIME message is sent to the user.

Business Rules
1. Sender user should be logged in 2. Receiver user name should exist in system.

Use Case Identification and History	
Use Case ID:	14
Use Case Name:	Recall Message
End Objective:	A message is recalled
User/Actor:	Any user
Trigger:	User
Frequency of Use:	Frequent

Preconditions
Prattle is running, and chatter is connected to it, and user is logged in

Basic Flow		
Step	**User Actions**	**System Actions**
1	User enters command RECALL <message id> < user/group>	System recalls the sent message.

Exception Flow	
1	If sender user is not logged in, error message is displayed
2	If message id does not exist, a fail message is sent

Post conditions
1. Message get recalled.

Business Rules
1. Sender user should be logged in

Use Case Identification and History	
Use Case ID:	15
Use Case Name:	Subpoena Login
End Objective:	Agency logs in for a subpoen
User/Actor:	Agency
Trigger:	User
Frequency of Use:	Frequent

Preconditions
Prattle is running, and chatter is connected to it,

Basic Flow		
Step	**User Actions**	**System Actions**
1	User enters command SUBPOENA <subpoena id>	System verifies if the subpoena id is presents, if yes then it logs in the connection for subpoena. And send all messages to connected chatter of that subpoena.

Exception Flow	
1	If subpoena id is incorrect user get fail message

Post conditions
1. Agency get logged in for single subpoena

Business Rules
1. Subpoena id should be present in the system

Use Case Identification and History	
Use Case ID:	16
Use Case Name:	Create Subpoena
End Objective:	Subpoena is created
User/Actor:	Admin user
Trigger:	User
Frequency of Use:	Frequent

Preconditions
Prattle is running, and chatter is connected to it, and admin user is logged in

Basic Flow		
Step	**User Actions**	**System Actions**
1	User enters command SUBPOENA_GROUP < group name> < from date> <to date> SUBPOENA_USER < user1> <user2/”all”> <from date> <to date>	System creates the group subpoena or user subpoena and sends back an unique subpoena id

Exception Flow	
1	If user is not admin user, no privileges message is sent back.
2	If the user/ group does not exist fail message is sent

Post conditions
1. Subpoena get created in the system

Business Rules
1. User should be admin user 2. User/ group should exist in the database 3. From date should be before to date

Use Case Identification and History	
Use Case ID:	17
Use Case Name:	Parental Control
End Objective:	Parental control is turned on or off
User/Actor:	any user
Trigger:	User
Frequency of Use:	Frequent

Preconditions
Prattle is running, and chatter is connected to it, and user is logged in

Basic Flow		
Step	**User Actions**	**System Actions**
1	User enters command PARENTAL_CONTROL <on/off>	System turn on/off parental control for the user and sends back success message. All sent message to that user is filtered based if parental control for user is on.

Exception Flow	
1	If sender user is not logged in, error message is displayed

Post conditions
1. Parental control gets turned on or off

Business Rules
1. User should be logged in.