Function test:

1. Title:
   UI check:
    a. The spelling is correct.
    b. The content is correct.
2. User profile photo:
   UI check:
    a. If the user is using a personal picture, the picture can be shown correctly.
    b. If the user doesn't use a picture, check if the default image could be shown correctly.
    c. The photo should show the correct user who is writing the posts.
3. Dialog editor:
   UI check:
    a. There is empty content by default or some hint words.
    b. The dialog window is located beside the user's profile image.

   Actions check:
    a. Users can fill in words inside the dialog, including characters, numbers, symbols(such as @, $, %, etc.) and the 'enter key'.
    b. Users can input some emoji.
    c. Check the max amount of words that is allowed for a post.
    d. Check if the user inputs empty text and verify the system handles it correctly.
    e. Check the line wrap by filling in a very long word that extends one line.
    f. Fill in a long word that exceeds the total limit of a post allowed.
    g. Fill in many words that the total amount of characters exceeds the input limit.
    h. Check copy and paste words function works.
    i. Users could delete the input content.
    j. Check if the system supports 'Undo' and 'Redo' operation for text input.
    k. In the right down corner, the user can draw it to resize the dialog window.
4. Drag the images:
   UI check:
    a. There is an icon in front of the text.
    b. The text is correct saying 'Drag and drop or upload'
    c. There is an underline for word 'upload'

   Actions check:
    a. Drag one image into the dialog window and the image could be shown correctly.
    b. Multi-select many images and drag them into the dialog window and all the images could be shown correctly.
    c. Drag the images with different image type, such as pgn, jpeg, img, ect. All the image types should be shown correctly..
    d. Drag one image with a big size, such as 20M, see whether it could be shown correctly. Or check whether there is a limit for the image size.
    e. Drag some txt file and video file that is not supported into the message editor and see if there is some error message showing or at least the message editor is not crashed.
    f. Upload a file and verify if the system displays a progress indicator showing the upload status.

g. Cancel the upload process and check if the system cancels the upload and reverts to the original state.
h. Try to upload a corrupted file or a file of unsupported format and check if the system can displays an error message.
i. After dragging in the images, there should be no impact to the text message the user has input.
j. User could delete the uploaded images

5. Upload images:
UI check:
a. The text is correct saying ‘Drag and drop or upload’
b. There is an underline for word ‘upload’
Actions check:
a. After clicking the word ‘upload’, a system window should pop up to let the user select the files to upload.
b. The user could select one or multiple images at a time, and all the images should be shown correctly.
c. The user could click cancel in the pop-up window, which means select no images to upload, and all the images should shown correctly.
d. After uploading in the images, there should be no loss to the text message the user has input.
e. The user could delete the uploaded images.
f. All the uploaded images should be shown correctly.
g. System should only support the image type of files to upload.
h. System should support multiple types of images to upload.

6. Post button:
UI check:
a. there is a ‘Post’ button below the message input dialog box
Actions check:
a. Post button is clickable if some message or image is input.
b. Post button is not clickable if nothing is in the dialog box.
c. After clicking the Post button, no message or images are lost compared with the content the user has input.
d. After clicking the Post button, the message and images are not editable.
e. Users post several ‘Posts’ or ‘Articles’, check the latest posts should be shown on the top.
f. check if the system can provide appropriate error prompt messages when user input text in an incorrect or invalid format.

7. Write article button
UI check:
a. There is a pencil icon in front of the words
b. Content is ‘Write article’
Actions check:
a. Write article button is clickable
b. After clicking ‘Write article’, the interface should change from Posts to Articles. And the ‘Write article’ should change to ‘Write posts’.
c. If a user has filled in some words or images, after clicking the ‘Write article’ button, the system should give the user a warning message that the filled content will be lost.

d. Should support back and forth switching the interface.

Security Test:

1. XSS Attack Check:
   Input text containing JavaScript code and ensure that the interface properly prevents cross-site scripting (XSS) attacks.
2. Image Upload Format Validation:
   Attempt to upload different types of files, including files other than images (such as .exe, .html, etc.). Ensure that only allowed image types can be uploaded.
3. File Size Limitation:
   Attempt to upload large files and ensure that the interface correctly limits file sizes to prevent abuse and denial-of-service (DoS) attacks.
4. Image File Name Security:
   Ensure the interface handles image file names containing special characters or malicious code.
5. Content Filtering:
   Ensure the interface filters or locks sensitive or inappropriate content.
6. Image Upload Path Isolation:
   Attempt to upload image files and modify the file path. Ensure that the system properly isolates files uploaded by users to prevent directory traversal attacks.
7. Error Handling and Exception Cases:
   Verify the interface handles network interruptions or upload failures properly.
8. Authentication and Authorization:
   Ensure that the system correctly verifies user identity and restricts access to functionalities based on permissions.
9. Encryption and Data Transmission Security:
   Monitor whether secure encryption protocols (such as HTTPS) are used during data transmission and whether sensitive data is properly encrypted to prevent data leakage or theft.

Performance Test:

1. Concurrent User Load Testing:
   Simulate multiple users accessing the interface concurrently to determine system capacity.
2. Response Time Testing:
   Evaluate system response times to ensure they meet acceptable thresholds.
3. Throughput Testing:
   Test the system's ability to handle a high volume of requests within a specific timeframe.
4. Resource Utilization Testing:
   Monitor system resources (CPU, memory, disk) under load to identify bottlenecks.
5. Cache Efficiency Testing:
   Assess the effectiveness of caching mechanisms in improving system performance.
6. Stability Testing:

Verify system stability under sustained load to detect potential issues like memory leaks.

7. Data Volume Testing:

   Increase data volume to assess system performance under large datasets.
8. Load Balancing Testing:

   Ensure load balancing distributes traffic effectively across servers.
9. Long-Term Stability Testing:

   Validate system stability over extended periods, such as 7*24 hours or more.
10. Database Performance Testing:

    Evaluate database performance, including I/O speed and transaction processing.

Other Testing Areas:

1. Cross-Platform Compatibility Testing:

   Assess performance across different operating systems and browsers for consistency. Such as Windows and MacOS.
2. Usability Testing:

   Evaluate the UI's ease of use, and UI design to ensure users can easily use the system.
3. Localization Testing:

   Validate the system's adaptation to different languages, time zone and regions.