Testing Plan

Guidelines:

- We will have an integration test for every week.
- One person is in charge of each integration test, and he is responsible to write a test report for an integration test.
- Everyone has to test all functions of his module.
- After individually testing code, the code must be reviewed by another member.
- If no bug is found in peer review, it will be merged to main repository.
- We will have an extra meeting if there is 10 or more open bugs.

Bug tracking agreement:

- If someone finds a bug during peer code review, write a brief report which states under what circumstances the bug is found and suggested solution if possible.
- The bug report should be submitted to "issue" in github.
- The author of the code is responsible to fix reported bugs.
- Once a bug is fixed, test and review code again.

Tools for bug tracking:

- github: post a bug report at issue section.
- logging function: implement logging function on the server to trace where a bug occurs. (everyone must use the same logging module which will be created at the first week).
- documentation: All code must contain documentation to help peer reviewing. Documentation must describe what the code is supposed to do briefly.

Unit testing and peer review:

Unit Testing:

When: each time one finishes creating a feature
Object: a module works as it is supposed to do.
Procedure:

Check the requirement and assumptions.
Write test harness if applicable.
Test the module so that it can produce required output, or result.
Test all equivalence classes as input.
If any bug is found, one should attempt to fix it before submitting a report.
Upload test report (and test harness) on our github repository.

Peer Review:

When: one finishes individually testing his code
Object: review other's code and find errors
Procedure:

Get other's code from github.
Go through the code to find errors.
Read test report for that code.

- if you think there should be more test cases, add and try them.

Try key test cases.
Report bug if there is any; otherwise merge it to main repository.

Testing Plans:

First Integration Test:

When: the end of the first week
Object: test interactions between a server, GUI and database.
Procedure:

1. Test the server
-Run the server.
-Send request from a web browser and observe its behaviour.
-open every page to check request handling.
-weight test: refreshing the page rapidly.
-Check if the server's log files are generated correctly.

2. Test GUI
-Check that all the widgets work correctly.
-Confirm that the interaction of buttons, icons, labels, menu, tabs, and text boxes work correctly.
-Compare actual web pages to mock ups and see if there is any problem.

3. Test discussion space
-Create a topic on discussion space.
-test exceptional case: no message in the topic body.
-create a topic which has more than 500 words.
-Check the topic is visible to others
-Edit the topic, and check if the change is reflected
-Post comments on the topic created above
-Input empty comment and huge comment (more than 3000 words).
-Edit the comment
-Confirm that the self-regulating functionality works well
-Delete comments
-Delete topics.
-Confirm that all information provided in the GUI is consistent with the data in the database.

Second Integration Test:

When: the end of the second week
Object: document page, searching, and commenting function
Procedure:

1. Test instant message

-Send a message
- test a long message (more than 100 words).
- test an empty message
- test a message that contains some characters that are encoded differently

-Observe the instant message window from another browser and check that the information sent is consistent with the information received.

2. Test user profile
-Add a new user profile to the database
-Check the database and see if the data is correctly stored into the database

3. Test scheduling
-Create a new event
-Modify the event
-input a long description (more than 100 words)
-Delete the event and check that every event in the calendar is consistent with the information stored in the database

4. Test Searching
-Search content by keywords
- test both found and not found cases
- Type a huge keyword (i.e, more than 200 words)

5. Test previous features
-Test features which are integrated in the first integration test

Third Integration Test:

When: the end of third week
Object: test memo, uploading files, and user log in functions
Procedure:

1. Test media functions
-Open the media page
-Upload a media and file
-Select and download any media file and check if it appears correctly.
-Search media by keywords
- make sure test both found and not found cases

3. Test user log in
-Set up a user account (ID, password and display name).
-Use the account to log in the system
-Check if the name is displayed correctly, and session remains until the browser is closed

4. Test previous features
-Test features which is integrated at the first and second integration test

Done

We are done the first iteration when the following are complete:

- Navigation works correctly thoughout all implemented features of the system
- There are no errors logged after running every test again
- able to add/remove users, and edit the education schedule.
- All authorized users can log on, add and comment on discussions, post videos, and view the schedule, discussions, and all media.
- The search works well enough to be able to search for keywords and tags
- Users can use the instant message to communicate each other