WhootChat

CEN 4010 Principles of Software Engineering Spring 2021

Milestone 4 Beta Launch

Team 10
Low Orbit Programmers

Matthew Vohland - mvohland2017@fau.edu - Product Owner

Darion Chong -dchong2019@fau.edu - Back-End Developer/Scrum Master

Redrick Horton-Schittko -rhortonschit2017@fau.edu - Back-End Developer

Trinity Carnegie - tcarnegie2017@fau.edu - Front-End Developer/Github Master

Aicsa Machado - machadoa2016@fau.edu - Front-End Developer

Florida Atlantic University
Dr. Shihong Huang
shihong@fau.edu

April 13, 2021

2.2 Product summary

WhootChat.com

Major Features

1. Users should be able to create an account
    - Every user should be able to create an account by providing username and password. Once the account is created, the user should be able to login and log out. (Fau email required)
2. User should be able to enter information on profile
    - Users should be able to enter their bio, major, year of graduation,and school they attend.
3. User should be able to upload multimedia
    - Users should be able to upload images, videos, files, and documents to the Chat Tab
4. User should be able to search any subject/major related items on the Search Bar
    - For example, users can search for notes provided by other students (willingly) on a specific subject.
    - Another example, students can search for usernames that correspond to students in their classes.
5. System should be able to provide “School Tab” section
    - Section where students can find information about classes, majors, TA’s and professors.
6. System should be able to to provide “Chat Tab” section
    - Section where students can interact with each other and upload multimedia like images, videos, files, and documents.
7. System should be able to provide a Search Bar
    - Students can type after clicking on the search bar and look for class notes as well as usernames.
8. System should provide a Security Management Module that helps reset passwords

In order to market to our target audience, FAU Students and Alumni, we would like to create a partnership with FAU. Doing this will greatly increase our chances of delivering a superior website and allow current FAU students to build on the current website.

A more direct approach to marketing would be to advertise homework and tutoring help on FAU bulletin boards throughout the campus. This includes passing out flyers to all clubs, dorms, as well as fraternities and sorority houses.

Another option to jumpstart our website is marketing through social media. There are cheap options to market our website to only local locations.

Marketing using social media, flyers, and the possibility of a partnership with FAU would be the least expensive and most effective way to reach our target market.

Link to Site http://whootchat.epizy.com/

2.3 Usability test plan – maximum 2 pages

Usability Test Plan

Product under test: Chat Function for the website Whootchat.com. The Chat Function will allow you to connect in real time with current students and alumni at FAU. You will be able to communicate about anything unfiltered.

Objectives of test:

- Do people understand how to use the chat function?
- Do people like the flow of the chat function, usability, functionality, appeal?
- How often would they use the chat function while on the site?
- Why else would you use this function?
- How could the chat function be better? What would you add?
- Do you like the chat interface and is it easy to use?
- Is the chat function convenient for you?

Test Plan:

Starting point will be logging on to whootchat.com. The user, User1, will then navigate to the chat option. Once User1 has entered the chat, User2 will be in the chat. For testing purposes we will set up a chat between two users. User1 and 2 will be able to converse about their upcoming work for their classes. This conversation will last for a brief period of time in order to demonstrate the usability of the send and receive features of the chat function. Finally, User1 will then log out and exit the program.

Questionnaire form

Please answer the following question based on a scale from 1 to 5

(1 being unsatisfied and 5 being extremely satisfied)

1. How important is it that you use a chat feature when being tutored?
2. How important is it that you use a chat feature when discussing general class questions?
3. How important is it to access the chat feature 24/7?
4. How important is it that no ads are present when using the chat feature?
5. Rate your experience using the chat feature?
6. Rate the time you send a message?
7. Rate the usability of the chat feature?
8. How important is it that tutors are available at any time?
9. Rate the visual appearance of the chat feature?
10. How frequently do you believe you would like to use this feature?
11. How complex was the chat feature?
12. How easy was it to use the chat feature?

2.4 QA test plan – maximum 2.5 pages

For the same function you chose for the usability test, write AND execute a QA test plan (check class slides)

a) Create a formal QA test plan (consult QA class material). Basically, it has to contain:

1) Test objectives: max 0.5 pages
2) Hardware and software setup: max 0.5 page
3) Feature to be tested: max 0.5 page
4) Actual test cases: 3 test cases and results of testing them on your system: 1 page

You must provide a test plan and test summary in the format (e.g. form) allowing easy reading and analysis by management e.g. in a table format like presented in the lecture.

Suggested format for QA Table columns are: test #; test title; test description; test input; expected correct output; test results (PASS or FAIL for each tested browser)

5) Perform the testing as per plan above and record the results in a form above.
6) Apply the above test on 2 browsers of different type and record it in the above table

Software Project Test Plan

Chat Function

1.0 Introduction

In this presentation we will assess the live chat function for the website WhootChat.com. This chat function allows users to send and receive messages from other users.

2.0 Objectives and Tasks

2.1 Objectives

- Analyzing the performance time of the responses between the user and admin.
- Assessing the ease and usability of the chat feature.

➢ Identifying the user experience of the chat feature.

2.2 Task

A User will login to account and select the chat tab. Once the user is admitted into the chat they will begin a conversation with another user. Eventually, the first user will exit the chat and sign out.

3.0 Test Environment

3.1 Test Environment Setup

A valid network connection is required to test the software as well as utilizing an up to date version of the testers chosen web browser (i.e: Google Chrome, Microsoft Edge, Safari, etc).

4.0 Test Cases

Chat Functionality Test #1

Evaluates the qualifications of each objective by marking it as: PASS if the objective has been met or FAIL if it has not been met for both the Google Chrome web browser and Safari.

Web Browser	**Chat feature is easy to use**	**The time between message sending and receiving is at an acceptable rate**	**Chat feature is convenient**
Google Chrome	PASS	PASS	PASS
Safari	PASS	PASS	PASS

2.5 Code Review

By now you should have chosen a coding style. In the report state what coding style you chose.

Choose the code (substantial portion of it) related to the feature you used for QA and usability test. You need to submit an example of the code (or part of it – 2 pages or so MAX) for its function to be peer reviewed, and document this as follows:

1) One team member should submit code to other team member(s) for peer review.
2) Peer review should be performed by other group member(s) (1 review is OK).
3) Peer review is to be done by e-mail and comments are to be included in the code
4) Submit listing containing the peer review and commented code and communication related to this in your Milestone 4 document

Important: It is critical that code reviews are friendly and helpful, intended to help and education, and not to criticize. It is strongly suggested that you use peer review in the development of the whole system.

Code Peer Review for Chat Function written in the language of HTML, CSS, and PHP taken from our PHP file.

```
<?php
session_start();
if(isset($_GET['logout'])){
  //Would be beneficial if there was an entrance message in the chat MV
        $logout_message = "<div class='msgln'><span class='left-info'>User <b class='user-name-left'>". $_SESSION['name'] ."</b> has left the chat session.</span><br></div>";
  file_put_contents("log.html", $logout_message, FILE_APPEND | LOCK_EX);
//names could be more simple  MV
  session_destroy();
  header("Location: index.php"); //Redirect the user to chat Landing page
}
//Enter name in Chat Function
if(isset($_POST['enter'])){
  if($_POST['name'] != ""){
    $_SESSION['name'] = stripslashes(htmlspecialchars($_POST['name']));
  }
```

```
else{
    echo '<span class="error">Please type in a name</span>';
}
if(!isset($_SESSION['name'])){
    loginForm();
}
else {
?>
    <div id="wrapper">
        <div id="menu">
            <p class="welcome">Welcome, <b><?php echo $_SESSION['name']; ?></b></p>
            <p class="logout"><a id="exit" href="#">Exit Chat</a></p>
        </div>
    <div id="chatbox">
        <?php     //check if user is typing, if so then post comments when submitted
        if(file_exists("log.html") && filesize("log.html") > 0){
            $contents = file_get_contents("log.html");
            echo $contents;
        }
        ?>
        </div>
    <form name="message" action="">
            <input name="usermsg" type="text" id="usermsg" />
            <input name="submitmsg" type="submit" id="submitmsg" value="Send" />
        </form>
    </div>
                                                    <script    type="text/javascript"
src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script type="text/javascript">
        // jQuery Document linked to jQuery library
        $(document).ready(function () {
            $("#submitmsg").click(function () {
                var clientmsg = $("#usermsg").val();
                $.post("post.php", { text: clientmsg });
                $("#usermsg").val("");
```

```
            return false;
        });

        function loadLog() {
            var oldscrollHeight = $("#chatbox")[0].scrollHeight - 20; //Scroll height before the request
            $.ajax({
                url: "log.html",
                cache: false,
                success: function (html) {
                    $("#chatbox").html(html); //Insert chat log into the #chatbox div
                    //Auto-scroll with each postings
                     var newscrollHeight = $("#chatbox")[0].scrollHeight - 20; //Scroll height after the request
                    if(newscrollHeight > oldscrollHeight){
                                $("#chatbox").animate({ scrollTop: newscrollHeight }, 'normal'); //Autoscroll to bottom of div
                    }
                }
            });
        }
        //check if user wants to terminate session
         setInterval (loadLog, 2500);
         $("#exit").click(function () {
            var exit = confirm("Are you sure you want to end the session?");
            if (exit == true) {
            window.location = "index.php?logout=true";
            }
         });
      });
    </script>
  </body>
</html>
<?php
```

Code Review: The code looks really good. You used comments properly so it is easy to follow along with code code and know what certain functions do. I did notice however that for your user logout function. It only displays when a user has left the live chat, it would probably be useful to also add a function that allows the site to tell the user when a person has entered the chat so that they know they are not in the chat by themselves. Also the (header Location:"index.php") should be set to the main landing paige and not have the user sent back to the beginning of the live chat tab.

'''

On Tue, Apr 13, 2021 at 9:45 P[redacted]> wrote:

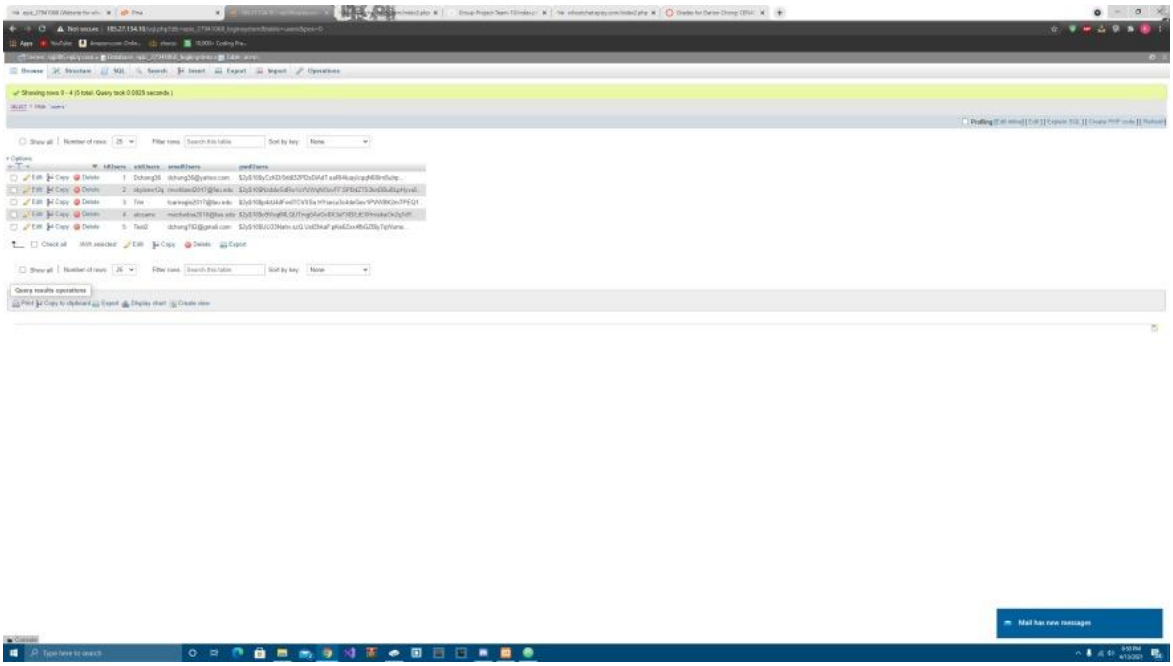
```php
<?php
session_start();
if(isset($_GET['logout'])){
  //Would be beneficial if there was an entrance message in the chat MV
  $logout_message = "<div class='msgln'><span class='left-info'>User <b class='user-name-left'>". $_SESSION['name'] ."</b> has left the chat session.</span><br></div>";
  file_put_contents("log.html", $logout_message, FILE_APPEND | LOCK_EX);
//names could be more simple  MV
  session_destroy();
  header("Location: index.php"); //Redirect the user to chat Landing page
}
//Enter name in Chat Function
if(isset($_POST['enter'])){
  if($_POST['name'] != ""){
    $_SESSION['name'] = stripslashes(htmlspecialchars($_POST['name']));
  }
  else{
    echo '<span class="error">Please type in a name</span>';
  }
  if(!isset($_SESSION['name'])){
    loginForm();
  }
  else {
```

2.6 Self-check on best practices for security – ½ page

1) List major assets you are protecting

- Passwords
- Data
- Email address

2) Confirm that you encrypt password in the DB



3) Confirm Input data validation (list what is being validated and what code you used) – we request that you validate search bar input;

For the testing of the Live Chat feature, our validations relied on external factors such as convenience, performance, and design. We validated it by having each of the group members create an account and test the feature with each other.

2.7 Self-check: Adherence to original Non-functional specs

Copy all original non-functional specs as in high level application document published at the very beginning of the class and then for each say **DONE** if it is done (which is expected and required); **ON TRACK** if it is in the process of being done and you are sure it will be completed on time; or **ISSUE** meaning you have some problems and then explain it.

Note: you must adhere to all original non-functional specs as published in the original high-level specification document. Failure to do so may cause reduced grade

List of non-functional requirements.

1. Storage Capacity
    a. Volume of data the system will persist at during run time ON TRACK
    b. Growth with user growth and usage ON TRACK
2. Security
    a. Login DONE
        i. User or Admin DONE
    b. Inactivity timeouts? ON TRACK
    c. Data back-up? ON TRACK
3. Audit/Maintenance
    a. File Characteristics - size before, size after, structure ON TRACK
    b. Time stamps for uploads and downloads as well as system maintenance. ON TRACK
4. Performance
    a. Response time, Browser refresh ON TRACK
    b. Processing times ON TRACK
5. Usability/Reliability
    a. Ability to perform its required function DONE

b. Response time/Processing times DONE

6. Recoverability
    a. Back up frequency ISSUE: No back-up recover system has been put into place for site
    b. Documentation ISSUE: No documentation for recoverability has been made
    c. User uploaded Documents ON TRACK
    d. Training Material ON TRACK
    e. Chat documentation ON TRACK

3 Submission

- Team lead submit Milestone 4 document to Canvas by due date
- Record a short demo of the current status of your product and post it on YouTube. List your YouTube URL here.

  Youtube Channel: https://youtu.be/Ys4fR4ZCRjw

4 Grading criteria

Your document needs to be well-written, well-organized (formatted) and reads well. Grading is based on cohesiveness and completeness.

1) Title page 10 points

2) Product summary 10 points

3) Usability test plan 20 points

4) QA test plan 20 points

5) Code review 20 points

6) Best practice for security 10 points

7) Non-functional requirements 10 points

Total: 100 points