CEN 4010 Principles of Software Engineering Spring 2021

Milestone 4: Beta Launch and Final Project Reviews
Group 11: DIVOC Forum

Team Members	Emails	Roles
Carlos Salazar	csalazar2018@fau.edu	Team Leader, Scrum Master, Back-End Developer
Zachary Goldstein	zgoldstein2018@fau.edu	Product Owner, GitHub Master, Back-End Developer
Alena Krause	akrause2017@fau.edu	Front-End Developer, Back-End Developer
Brandon Que	bque2018@fau.edu	Front-End Developer
Shannon Soulard	ssoulard2018@fau.edu	Front-End Developer

13 April 2021

Version History:

1.01	Document Template	11 February 2021
1.02	Initial Draft	15 February 2021
1.03	Final Draft	16 February 2021
1.04	Milestone 3 Adjustments	6 April 2021
1.05	Milestone 4 Adjustments	13 April 2021

2.2 Product summary

DIVOC combines elements of social media platforms as well as elements from resource forums to create an educational and community-based site all in one. DIVOC achieves this by having an online community of people to share information, educate each other, and provide emotional support during these difficult times of being socially distant from others. DIVOC allows signed-out users to:

1. create an account,
2. Search through posts using the search bar,
3. View basic user profiles.
4. view posts but not create posts,
5. nor interact with posts,

While signed in users can:

6. Comment on posts, create posts, and like posts,
7. Create a post within a specific topic,
8. Give the post a title and body of text,
9. Interact with posts by commenting and giving the post a “like”,
10. Search through posts using the search bar,
11. View basic user profiles,
12. Edit personal profiles.

Click on the URL below to see our website in action!

https://lamp.cse.fau.edu/~cen4010_s21_g11/forum-topic-prototype/www/home.html

2.3 Usability test plan

1) Test objectives: 0.5 page

> The objective of this test is to uncover the current usability of the DIVOC Forum in its current state (beta 4/13/21). One of the most important features of the DIVOC Forum is creating posts. This test is designed to pinpoint where our post contribution feature lies in terms of intuitiveness.
>
> Group 11 hopes that this test will answer questions such as:
>
> - Can users successfully navigate the topics page of the DIVOC Forum?
> - Can users successfully create posts that contribute to a given topic?
> - Will users be able to easily identify which button allows for post contribution?
> - Will users be able to successfully input their post content in the pop-up window?
> - Will users be able to easily sign in from the topics page?
> - Will users be able to easily view their contributed posts?

2) Test plan: System setup, starting point, task to be accomplished, who is the intended user, completion criteria, URL of the system to be tested. 3/4 page

The DIVOC Forum is intended for anyone to use, as the main goal is to allow people to share their thoughts about COVID.

System Setup:

- Testers will be given a link to the DIVOC Forum's topic page as well as the link to the follow up questionnaire.

Starting Point:

- Testers should start on the tester instructions page. They will be given instructions linked here.

Tasks to be Accomplished:

- From the topics page, testers will be asked to create an account using either google sign in or by registering with an email and password. Testers will be asked to create a post in three distinct topics of their choice and subsequently view said post on the forum. No additional information or instructions will be given to the testers to ensure that the results of the test reflect the site's true usability.
- Posts should follow the format listed below:
    - Title: [Tester Number] (numbers will be assigned by the development team)
    - Post: "I have contributed to the [TOPIC NAME] topic" (where [TOPIC NAME] should be replaced by the name of the topic the tester is contributing to).

Intended Users:

- The intended users of the DIVOC Forum consist of the general public. In order to replicate this, the chosen testers are all family members and friends of the development team (as well as some members of the development team). These testers come from a plethora of different backgrounds with varying degrees of skill when it comes to technology use.

Completion Criteria:

- Testers should be able to:
    - Create an account on the topics page.
    - Navigate between the different topics.
    - Locate the button that allows a user to contribute to a post.
    - Contribute a post to the three distinct topics of their choice.
        - Post content should follow the style specified in the instructions.
    - View their contributed posts to a topic.

3) Questionnaire form: 3 Likert scale questions, in a form easy to be used by reviewers (check class slides). 3/4 page

The usability questionnaire for the DIVOC forum can be found here, with the results being recorded on this spreadsheet.

The questionnaire consists of four Likert Scale questions:

1. The post contribution feature is easy to use.
2. The user interface when creating a post is intuitive (pop-up window).
3. Viewing my uploaded post was easy.
4. The topics page is easy to navigate.

Chosen Testers:

Tester Number	Name	Results
1	David Kang	Completion Successful Observations: User was confused about what the different topics were and where to find them.
2	Odalys Salazar	Completion Successful Observations: User found the website easy to navigate through. Had no problem locating the topics and contributing a post to each topic.
3	Terrence Charitable	Completion Successful Observations: User had no issues navigating through the subforums and had no issues contributing a post to the topics of their choice.
4	Brandon Que	Completion Successful: Observations: User found the website easy to navigate through. Users had no issues making posts and submitting.
5	Cristina Jimenez	Completion Successful Observations: User had no issues performing the given task.

Form Responses:

	The post contribution feature is easy to use.	The user interface when creating a post is intuitive (pop-up window).	Viewing my uploaded post was easy.	The topics page is easy to navigate.
1	Strongly Agree	Strongly Agree	Strongly Agree	Neutral
2	Strongly Agree	Strongly Agree	Strongly Agree	Strongly Agree
3	Strongly Agree	Strongly Agree	Strongly Agree	Strongly Agree
4	Strongly Agree	Strongly Agree	Strongly Agree	Strongly Agree
5	Strongly Agree	Strongly Agree	Strongly Agree	Strongly Agree

2.4 QA test plan

1) Test objectives: max 0.5 pages

The objective of this test is to uncover code defects in the DIVOC Forum. One of the most important features of the DIVOC Forum is creating posts. Group 11 hopes that this QA test will answer questions like:

- Can users that are not signed into the forum make any posts?
- Are users able to make any posts that have no content (i.e. no title or text for the post)?
- Are users able to exceed the character limit for the post title (max 100 characters) and post text (max 1000 characters)?

Making sure that the questions above pass their expected correct output in the actual test cases are essential. Users who are not signed in should never be allowed to make a post; posts with no content should not be accepted; posts' character count is limited to maintain forum organization

2) Hardware and software setup: max 0.5 page
Hardware: Any PC that can run browsers such as Chrome, Edge, Firefox, etc.

Software: Heading over to DIVOC's topic page which allows the users to sign in/create an account. Users will need to create an account in order to proceed with choosing a topic to create a post in (users will be able to view posts created by others even if they are not signed in). Once signed in after creating an account, the users will have full access to DIVOC's features such as viewing posts in the provided topics; creating a new post; viewing their own profiles which contain user specific information.

3) Feature to be tested: max 0.5 page

The feature that will be tested is making a post to contribute to the forum. This test will uncover any defects in this feature that users may encounter when making a post. Some test cases that will be used include:

- Testing if the user can make a post while not signed in.
- Testing if the user can post a contribution to the forum that contains no text or title.
- Testing if the user can exceed the character limit that is set for the post title and text

4) Actual test cases: 3 test cases and results of testing them on your system: 1 page

Google Chrome	Test Description	Test Input	Expected Correct Output	Test Results
Test Case 1	Test if the user can make a post while not signed in	Submit post without sign in	User should not be able to make post while signed out	PASS
Test Case 2	Test if the user can submit a post w/o any content in it	Have user submit a post with no content	Posts that do not have any content should not be accepted	FAIL
Test Case 3	Test if the user can exceed the character limit of a post	Submit a post exceeding character limits	Users should not be able to exceed the character limit for a post's title and body of text	PASS

Microsoft Edge (IE)	Test Description	Test Input	Expected Correct Output	Test Results
Test Case 1	Test if the user can make a post while not signed in	Submit post without sign in	User should not be able to make post while signed out	PASS
Test Case 2	Test if the user can submit a post w/o any content in it	Have user submit a post with no content	Posts that do not have any content should not be accepted	FAIL
Test Case 3	Test if the user can exceed the character limit of a post	Submit a post exceeding character limits	Users should not be able to exceed the character limit for a post's title and body of text	PASS

2.5 Code Review

1. Coding style

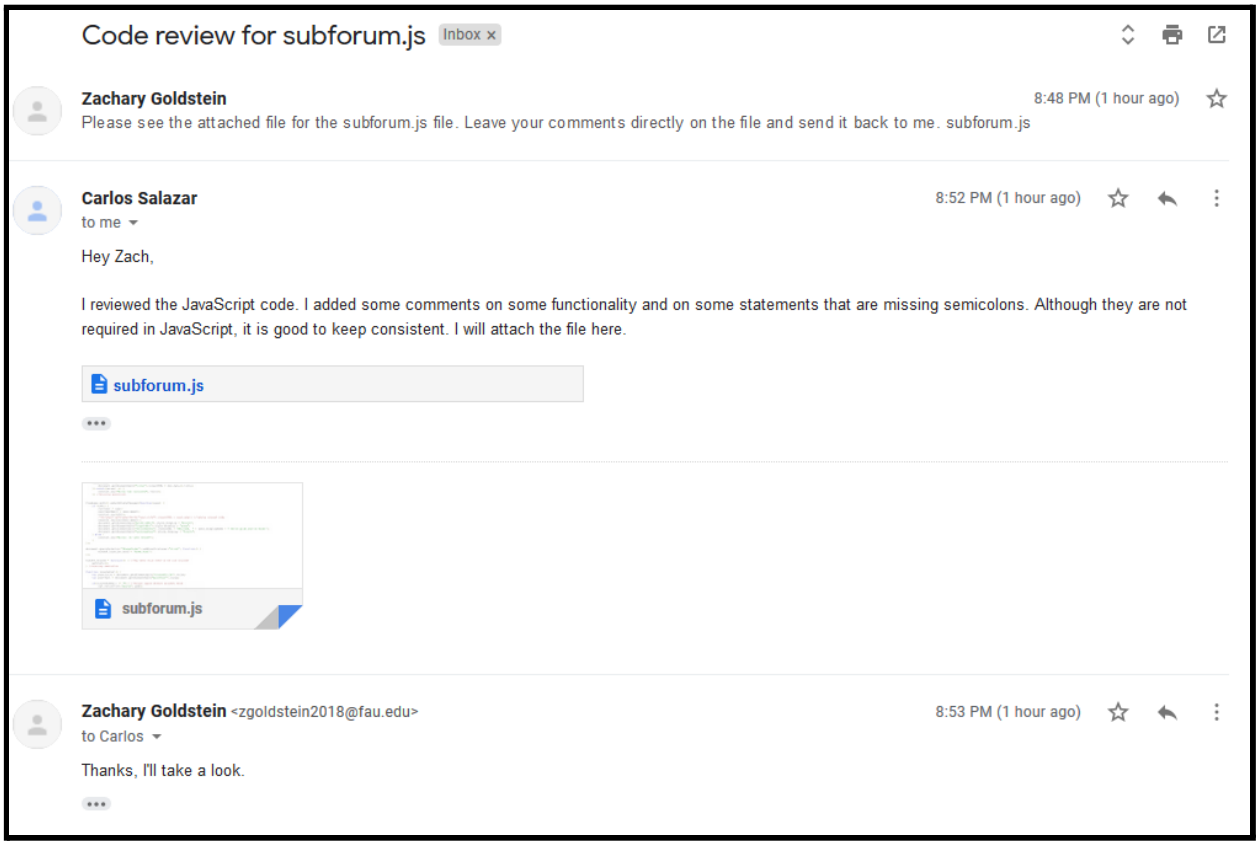
a. We do not require function headers or comments as we believe good code should not need comments. Any code that has comments is for clarity for code reviewing.
b. We used spaces instead of tabs
c. Lines were limited to 80 characters
d. CamelCase was followed for the naming of functions and variables.
e. Constants were used whenever possible to better optimize the code.
f. Semicolons were used for end of lines in the javascript code.
g. We used spaces before brackets. For example:

```
if(x) {
    doSomething;
}
```

Code review:

```
window.onload = function() { //why does this need to be ran onload?
    getPosts();
} //missing semicolon

function createPost() {
    var postTitle = document.getElementById("threadTitle").value;
    var postText = document.getElementById("postText").value;

    if(currUserEmail != ""){ //forgot space before bracket here
        ref.collection('posts').add({
            title: postTitle,
            text: postText,
            owner: currUser.email,
            createdAt: firebase.firestore.FieldValue.serverTimestamp()
        })
        .then((post) => {
            alert("Post Successful! Thank you for your contribution to
the DIVOC Forum");
            console.log("Document successfully written!");

            db.collection("users").doc(currUser.uid).update({
                posts: firebase.firestore.FieldValue.arrayUnion(post)
            }) //missing semicolon

            ref.update({
                numOfPosts: firebase.firestore.FieldValue.increment(1)
            }) //missing semicolon

            location.reload();
        })
```

```
        .catch((error) => {
            console.error("Error writing document: ", error);
        });
    }else{
        alert("Hey bud, please sign in to contribute to the forum!");
//change to more professional wording
        $("#threadModal").modal("hide");
    }

}

function getPosts() {
    ref.collection('posts').orderBy('createdAt', 'asc').get()
        .then((doc) => {
            doc.forEach(doc => {
                posts.push(doc);
            });
            renderPosts();
        })
        .catch((error) => {
            console.log(error);
        });
}

function renderPosts() {
    var postElem = document.getElementById("postCard");

    posts.forEach(doc => {
        const data = doc.data();
        var clonePostElem = postElem.cloneNode(true);
        clonePostElem.id = doc.id

        var cardTitle = document.getElementById("cardTitle");
        var cardText = document.getElementById("cardText");
        var cardUser = document.getElementById("cardUser");
        cardTitle.innerHTML = data.title;
        cardText.innerHTML = data.text;
        cardUser.innerHTML = data.owner;
        postElem.after(clonePostElem);
        postElem.style.display = "block" //hides model card
    })
}
```



2.6 Self-check on best practices for security

1. List major assets you are protecting
    a. We tried to limit saving any sensitive assets in our database. The only major assets we are storing and protecting are:
        i. User's email
        ii. User's password
2. Confirm that you encrypt password in the DB
    a. We are using Firebase authentication to encrypt user's passwords. We are hashing the user's password using the Scrypt algorithm. Scrypt is a computationally intensive algorithm that works by filling the user's password with "noise" or randomly generated numbers to increase the work time to increase work time. The user's password is never stored in plain text in the database, and will only ever contain the password hash.
3. Confirm Input data validation (list what is being validated and what code you used) – we request that you validate search bar input;
    a. When a user creates an account they are required to input an unused email. If they enter an email that is already in use it will give them an error and not allow them to create the account. We also require the user to input their password two times to confirm that they have typed in the right password.

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

1. Security
    a. Password hashing to ensure data security for users. **DONE**
2. Portability
    a. Website optimization for access on a personal computer or mobile device. **ON TRACK**
    b. Compatible with all modern browsers including Firefox, Safari, Chrome, and Edge. **ON TRACK**
3. Expected Load
    a. Limit the number of requests made by a single user to avoid server crashes due to high traffic. **ON TRACK -** handled by Firebase
4. Usability
    a. Information is available to any site visitor. **ON TRACK**
    b. Features such as commenting on, liking, and creating posts are limited to only those who possess an account. **ON TRACK**
5. Storage
    a. Data retained will be restricted to the capabilities of the Firebase Firestore Database. **DONE**
    b. Posts will have a 1,000 character limit. **DONE**
6. Accessibility
    a. Site is easy to use and intuitive for users of any age. **DONE**
    b. Aesthetically pleasing and organized UI. **DONE**
    c. Site will not require any specialized training for use. **DONE**

3 Submission

• YouTube URL: https://youtu.be/FCsE3ef-rR8