



Python Software Foundation

Sub Organisation: **EOS Design System**

Project :

**User Story - Improvements
and New Features**

By: Harshita Mangla

Personal Details	2
Abstract	3
Outline and Methodology	4
Code Contributions	13
Schedule	14
Previous Experience	16
Personal Projects	17
About Me	17
Other Commitments During Summer	18

Personal Details

Contact Details:

Name: Harshita Mangla

Email: manglaharshita@gmail.com

Timezone: Indian Standard Time (UTC+5:30)

Profiles:

Gitlab: [mharshita](#)

Github: [mharshita](#)

LinkedIn: [harshitamangla](#)

Portfolio: [mharshita.github.io](#)

Education Details:

College: Guru Jambheshwar University of Science and Technology

Degree: B.Tech in Computer Science and Engineering

Current Year: 2nd Year

Expected to Graduate: May 2023

Abstract

EOS User Story provides an interface to the users to share how they are using the product and relate to other users' stories and vote them up. It helps users tell developers how they are using the products and request new features, report bugs etc. The project aims to enhance the user experience by introducing new features and improve on the existing ones. These features are as follows :

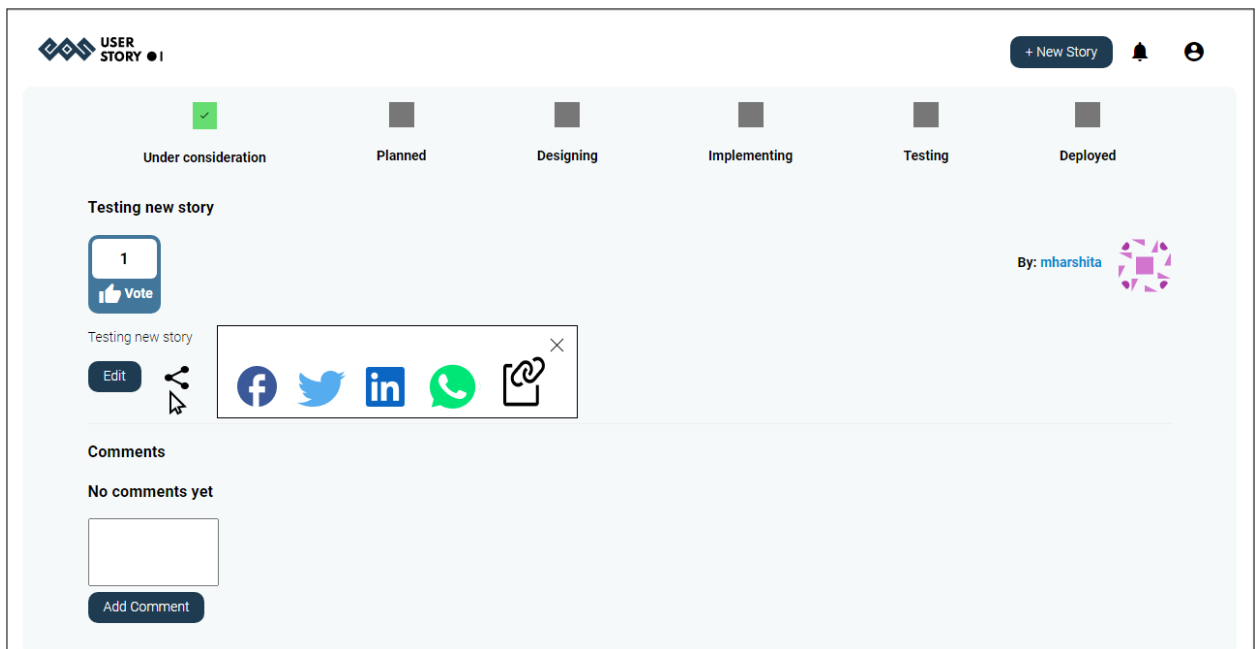
- **Generate a sharable link with one click for sharing stories with users across various platforms.**
This will allow the users to share the stories across various social media platforms like Facebook, Whatsapp, Twitter, and LinkedIn with a single click along with a template. With this functionality, users will also be able to copy the link to the story.
- **Search stories via the home page.**
Currently, whenever a user makes a new story, the title search functionality helps to recommend the already made stories. Similar functionality can be implemented for the home page which would help the users to search for a story right from the home page. Moreover, the current functionality helps to search for the title of the story, but this can be enhanced and searching can be implemented for story description and numeral inputs also.
- **Increase test coverage by writing more unit tests for edge cases.**
Test coverage can be increased by writing more tests for already existing functionalities like voting feature, attachments feature for stories (this is currently a bug I am working on), sorting feature and my stories and my account pages.
- **Request templates.**
Request templates will provide the functionality of providing pre-defined templates for products. It will help provide guidelines for things that have a standard in place already.
- **Personalized email notifications.**
This functionality will help the admins to send personalized email notifications to the users. For users, email notifications can be sent for sign-up, comments, and status updates for the stories. For admins, email notifications can be sent whenever a user uses bad words while creating stories and comments.
- **Allow users and admins to add priority labels to stories like 'High', 'Critical', and 'Moderate' while making a story.**
Currently, users can choose the priority of a story while creating a new story. This can be displayed in the form of tags on the home page.
- **Sorting of stories based on 'Important', 'Critical', and 'Moderate' prioritization labels.**
Functionality to provide sorting of stories on the basis of priority can also be implemented.
- **Allow the users to mention other users in the comments section of the story.**
This functionality will allow users to mention other users in the comments section of the story.

Outline and Methodology

- **Generate a sharable link with one click for sharing stories with users across various platforms**

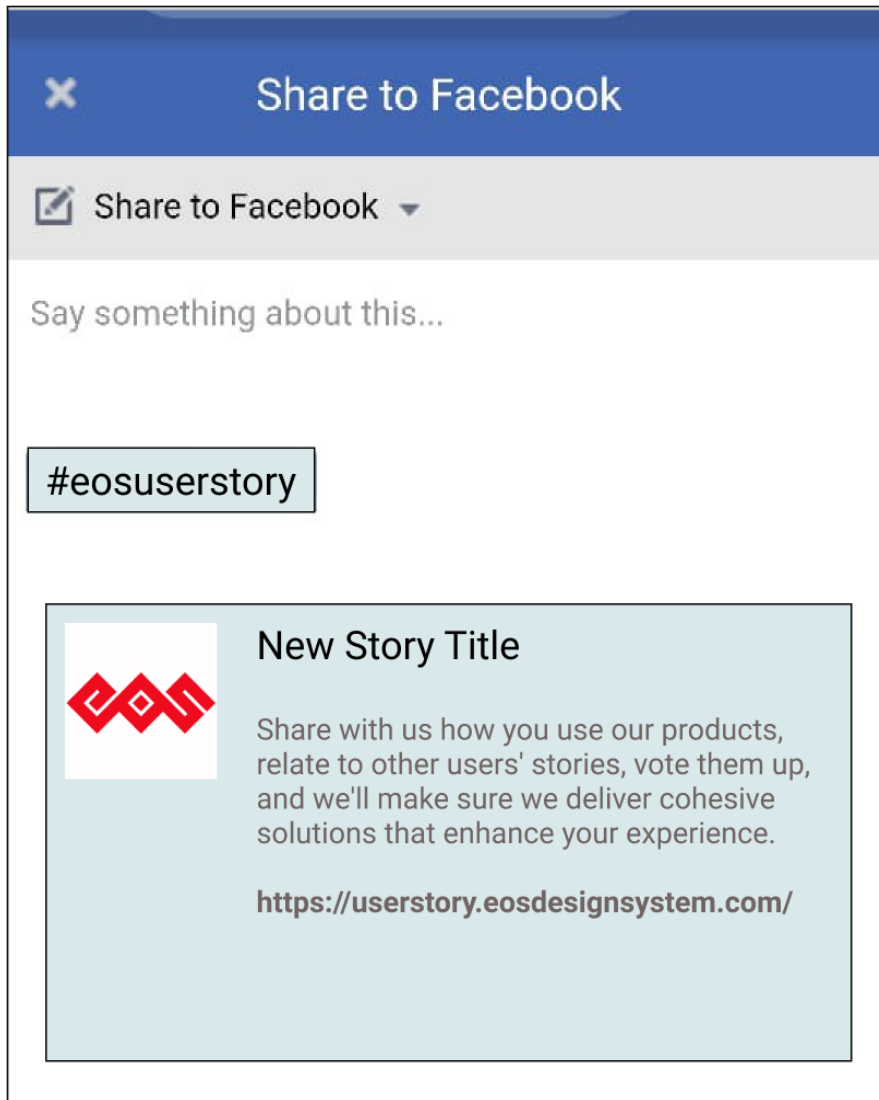
Currently, there is no way through which the user can share any story from EOS User Story on social media platforms. Hence, the idea is to introduce a *share button* that will allow the user to share link for the story across various platforms.

Prototype:



The share button will allow users to share the link to the story page to various social media platforms or simply copy the link of the story page to their clipboard. This can be achieved through [react-share](#) and [react-helmet](#) (already implemented in EOS User Story) libraries. The reason for using react-helmet is to provide a template to the social media post that will be sharing the story. Different templates can be made for different products. Example: If a user is sharing a story related to EOS Icons, a different template is shown and if a user is sharing a story related to EOS User Story, a different template can be shown like in the image shown here.

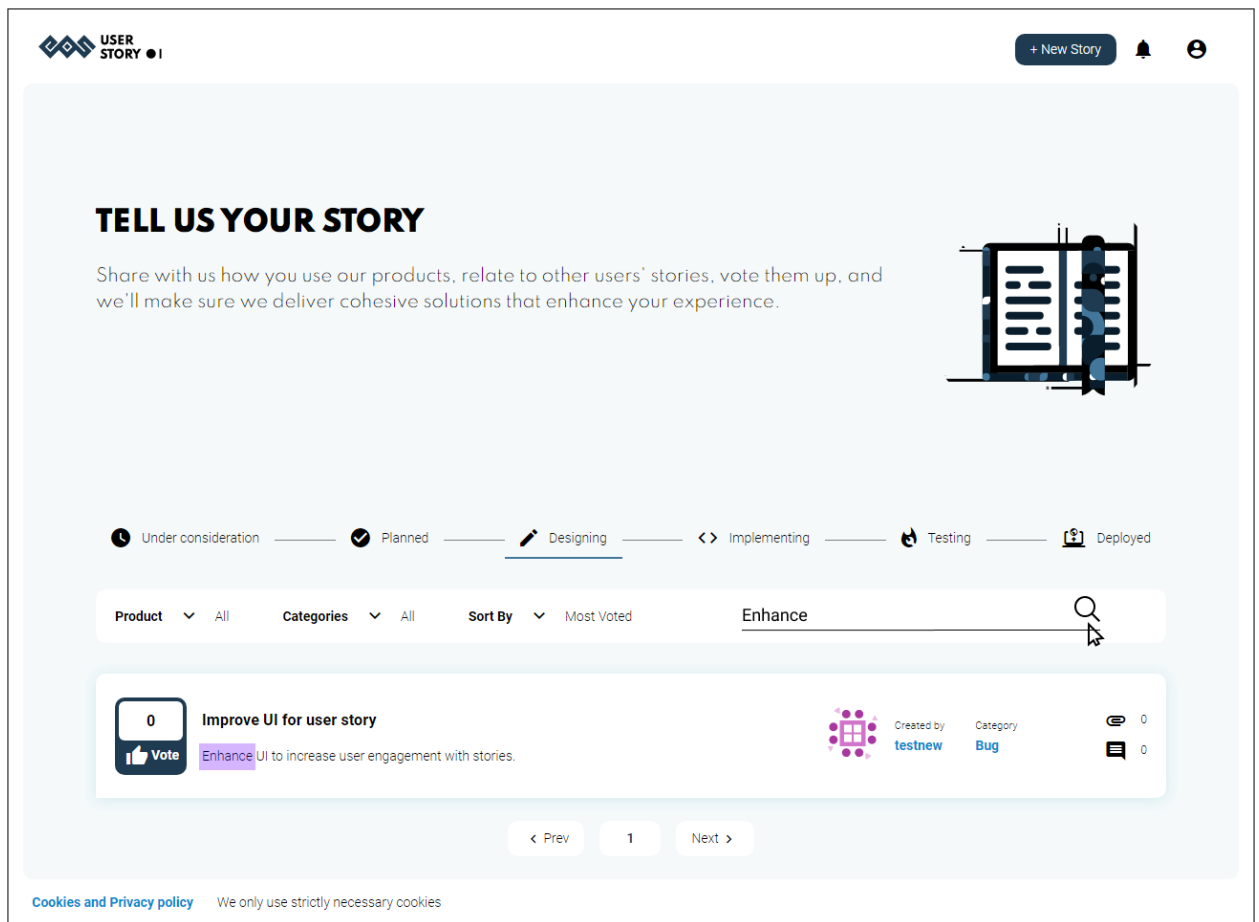
Example:



- **Search stories via the home page**

Implementation of search story feature from the home page of User Story. Currently, the search title feature is implemented whenever a user makes a new story. Current feature searches for the title of already posted stories. Now, for the home page, the existing functionality can be improved that will allow users to search from existing stories' title and description both. The feature can be further enhanced to search for numeric input for numbers present in the story title and story description.

Prototype:



• Request Templates

The request templates will provide the ability to configure predefined templates for certain products and help products provide guidelines for things that have a standard in place already. While making a new story, whenever a user selects the product, the description box will provide a template for that particular product. For a better user experience, a template would be provided by default but users can choose whether they want a template or blank description through a checkbox or dropdown or a toggle switch.

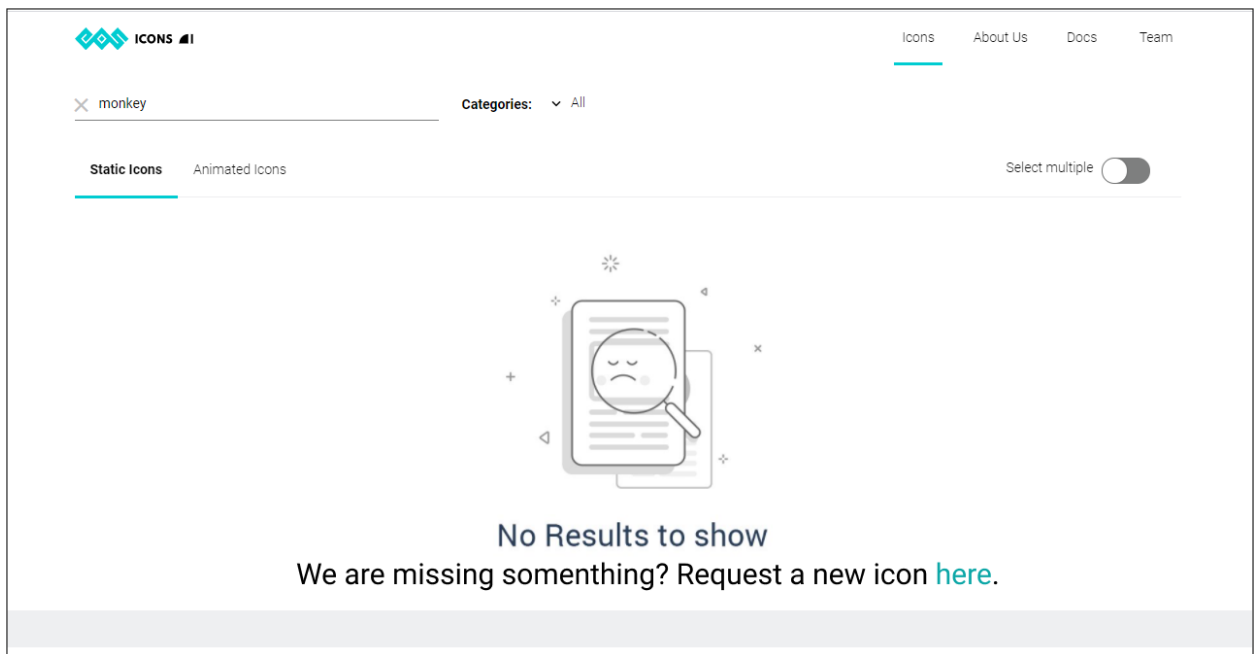
Example: Request Template for EOS Icons (New Icon Request) -

If a user wants to make a new icon request on EOS User Story platform, they will be provided with a template in the description of the new story that will ask name, image, category of the icon, icon will be static or

animated and a short description on how the icon can be used on the interface.

Along with this, on the [EOS Icons page](#), an option can be provided to the users to request a new icon if they are not able to find a particular icon. Links to request a new icon can be included in the footer and docs as well in a similar way.

Prototype:



The screenshot shows the 'New Story' form in the EOS system. At the top left is the 'USER STORY' logo. At the top right are a '+ New Story' button, a notification bell, and a user profile icon. The form fields are as follows:

- Title:** A text input field containing 'New Icon Request'.
- Product:** A dropdown menu with 'EOS Icons' selected.
- Category:** A dropdown menu with 'User_experience' selected.
- Priority:** A dropdown menu with 'Moderate' selected.
- Use Template:** A checked checkbox.
- Description:** A rich text editor with a toolbar containing 'Paragraph', bold (B), italic (I), link, bulleted list, and numbered list icons. The text area contains:
 - Icon Name -
 - Category of Icon -
 - Type of Icon (static / animated) -
 - Icon is used on the interface as -
 - Attach Icon image below.
- File Upload:** A dashed box containing the text 'Drag 'n' drop some files here, or click to select files'.
- Submit:** A dark blue button at the bottom center.

At the bottom left, there is a link for 'Cookies and Privacy policy' and a note: 'We only use strictly necessary cookies'.

- **Increase test coverage by writing more unit tests for edge cases.**

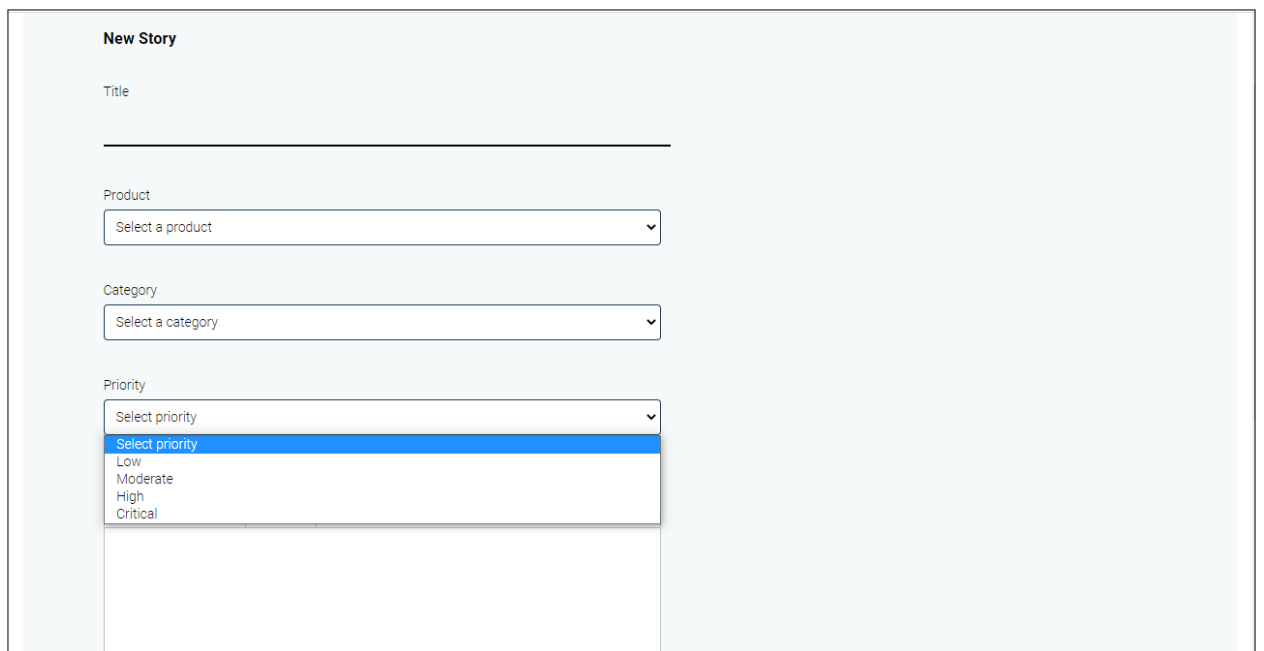
Writing more unit tests for edge cases for EOS User Story to increase test coverage. Tests in User Story are written using **Cypress**, **Mocha**, and **Chai**. Some of the areas where tests can be implemented for existing functionalities can be *voting feature* for stories (vote or unvote a particular story), *attachments feature* for new stories (this is currently a bug I am working on), *sorting feature* for stories. *Along with this, tests for the "My*

*Stories” page and “My Account” page are yet to be written. Further, tests are to be written for all the new functionalities that will be added. While working on tests, I will also be working on **bug fixes** in User Story. I have added a test for the voting feature and attached a video.*

[Click here to view the demo video](#)

- **Allow users and admins to add priority labels to stories like ‘High’, ‘Critical’, and ‘Moderate’ while making a story.**

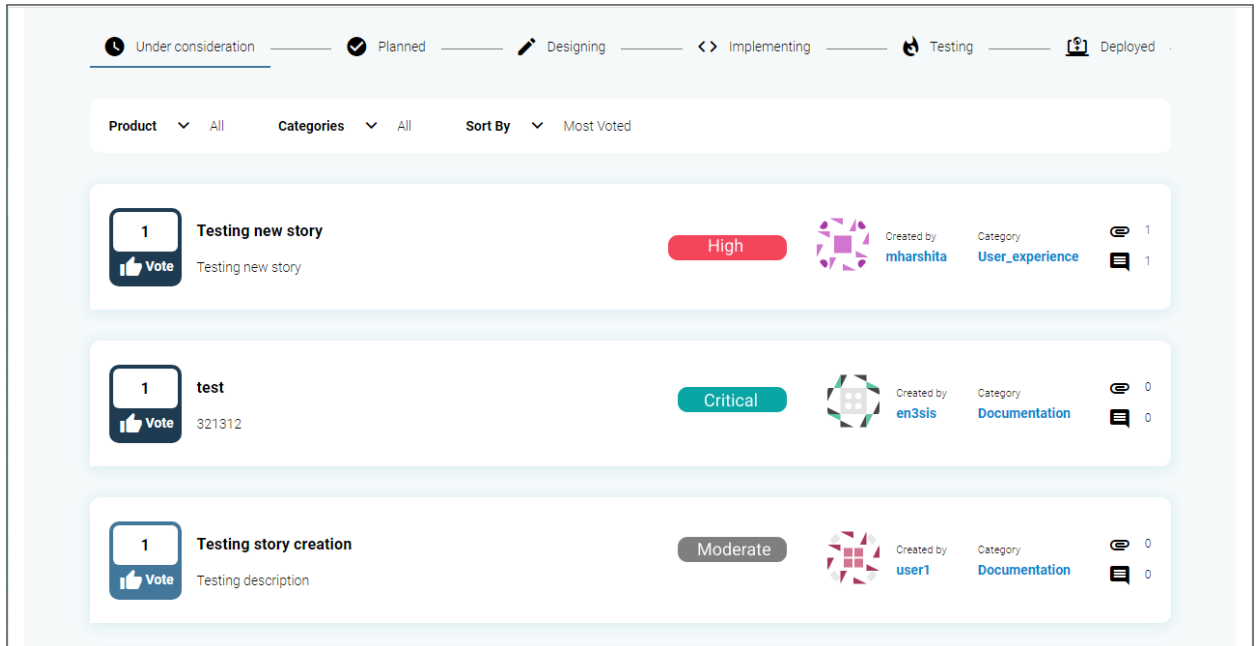
A new feature of letting the admins add priority labels can help developers and admins to prioritize the stories to work on. Currently, while making a new story, users and admins are asked to select the priority for the story but this is not displayed anywhere. Hence the selected priorities can be displayed in different colors in a way similar to the given prototype for the home page.



The image shows a 'New Story' form with the following fields:

- Title: A text input field.
- Product: A dropdown menu with the placeholder text 'Select a product'.
- Category: A dropdown menu with the placeholder text 'Select a category'.
- Priority: A dropdown menu with the placeholder text 'Select priority'. The dropdown is open, showing the following options: 'Select priority' (highlighted in blue), 'Low', 'Moderate', 'High', and 'Critical'.

Prototype:



- **Sorting of stories based on ‘High’, ‘Critical’, and ‘Moderate’ prioritization labels.**

Currently, the home page along with other pages offers sorting of stories based on Most Voted and Most Discussed but considering the previous feature, sorting can also be offered on the basis of priority.

- **Personalized email notifications.**

For Users:- Personalized email notifications can be sent to the user when the user signs up to user story or whenever someone comments on their story or when the story status changes or reaches deployed state. Users can also be given an option to invite other users by sending a personalized email invite through User Story.

For Admin:- Email notifications can be sent to the admins whenever a user uses bad words while creating stories and comments. The functionality will allow admins to block or give warnings to these users accordingly. To check for bad words, [bad words node module](#) can be used.

For giving email notifications for comments, we need to call the email service when the comment is created via POST/comments endpoint. For sending the email, [email plugin](#) will be used.

```

const emailTemplate = {
  subject: 'Welcome <%= user.firstname %>',
  text: `Welcome on mywebsite.fr!
  Your account is now linked with: <%= user.email %>.` ,
  html: `

# Welcome on mywebsite.fr!</h1> <p>Your account is now linked with: <%= user.email %>.<p>`, }; await strapi.plugins.email.services.email.sendTemplatedEmail( { to: user.email, // from: is not specified, so it's the defaultFrom that will be used instead }, emailTemplate, { user: _.pick(user, ['username', 'email', 'firstname', 'lastname']), } );


```

```

const { parseMultipartData, sanitizeEntity } = require('strapi-utils');

const Filter = require('bad-words');
const filter = new Filter();

module.exports = {
  async create(ctx) {
    let entity;
    if (ctx.is('multipart')) {
      const { data, files } = parseMultipartData(ctx);
      entity = await strapi.services.comment.create(data, { files });
    } else {
      entity = await strapi.services.comment.create(ctx.request.body);
    }

    entity = sanitizeEntity(entity, { model: strapi.models.comment });

    // check if the comment content contains a bad word
    if (entity.content !== filter.clean(entity.content)) {
      // send an email by using the email plugin
      await strapi.plugins['email'].services.email.send({
        to: 'paulbocuse@strapi.io',
        from: 'admin@strapi.io',
        subject: 'Comment posted that contains a bad words',
        text: `
          The comment #${entity.id} contain a bad words.

          Comment:
          ${entity.content}
        `,
      });
    }

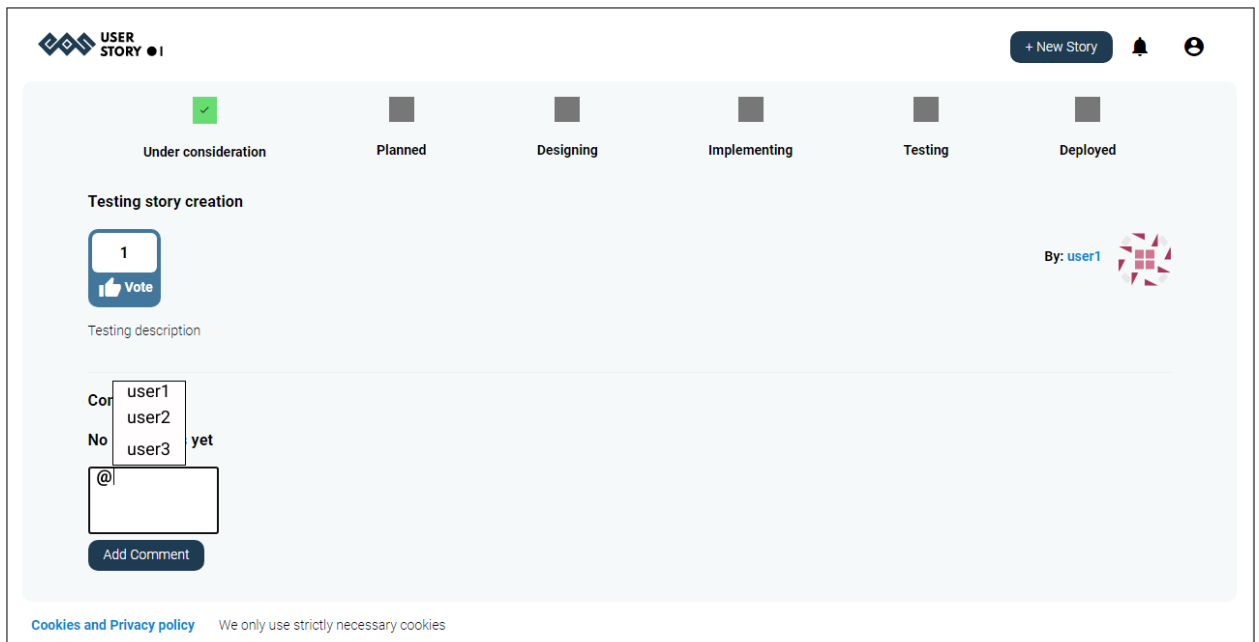
    return entity;
  },
};

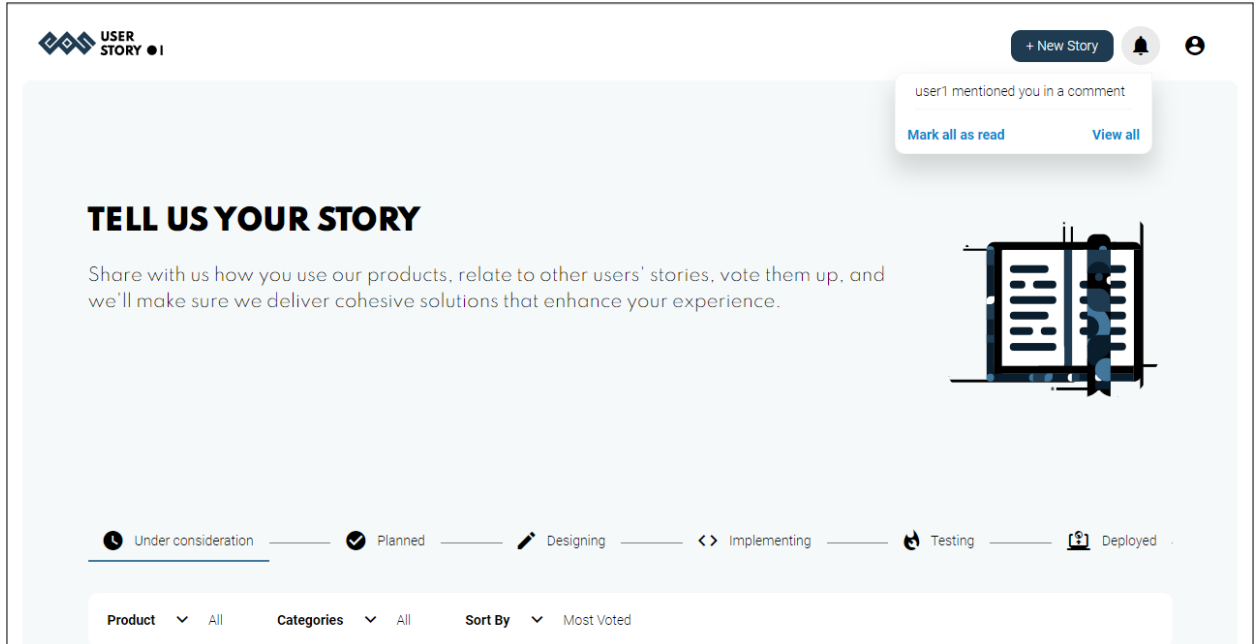
```

- **Allow the users to mention other users in the comments section of the story.**

The functionality will allow the users to mention other users in the comments section. The mentioned user will get the notification. This will be achieved using the [react mentions library](#). React mentions library supports both *user ID* and *username*. Whenever a user is mentioned, *react mentions library* wraps up the mentioned user as : `@[username][userId]` along with the text as (*Hey @[username][userId], How are you doing ?*). On mention, notification will be sent to the server with mentioned username in the user field and then fetched to display.

Prototype:





- **Working on Bug Fixes.**

Along with these functionalities, I will keep solving bugs in the pre community bonding period and also during GSOC.

I am looking forward to becoming a long time contributor and would love to keep contributing to EOS even after GSOC ends.

Code Contributions

Code Contributions made till 13 April, 2021 :

1. https://gitlab.com/SUSE-UIUX/eos-user-story/-/merge_requests/136
In this PR, I have added the display voters feature. This feature allows users to view the usernames of all the users that voted for that particular story.
2. https://gitlab.com/SUSE-UIUX/eos-user-story/-/merge_requests/137
Fixed bug - Display voters feature not working for my stories page.
3. https://gitlab.com/SUSE-UIUX/eos-user-story/-/merge_requests/131
Fixed bug - Error messages for signup form fields not showing correctly.
4. https://gitlab.com/SUSE-UIUX/eos-user-story/-/merge_requests/134
Fixed highlighting for selected tabs on my stories page.

5. https://gitlab.com/SUSE-UIUX/eos-user-story/-/merge_requests/141
Fixed bug - Improvements in Tests.
6. https://gitlab.com/SUSE-UIUX/eos-user-story/-/merge_requests/133
The PR allows the edit story feature to edit previously saved content and new content both.
7. https://gitlab.com/SUSE-UIUX/eos-user-story/-/merge_requests/129
Added 404 page that will be shown whenever a non-existent page is accessed.
8. https://gitlab.com/SUSE-UIUX/eos-user-story/-/merge_requests/124
Fixed bug - User Policy modal is modified.
9. https://gitlab.com/SUSE-UIUX/eos-user-story/-/merge_requests/119
Fixed bug - Responsiveness for sign-in and sign-up pages.

Link to all the Contributions made :

[Link to my code Contributions](#)

Schedule

Community Bonding Period (17 May - 7 June, 2021)

- Continue working on minor bugs and improvements.
- Spend more time learning about the workflow of EOS.
- Will get familiar with the community members and mentors and improve on the tech stack and timeline.
- I will also spend some time on university exams during this period.

Week 1 (June 7 - June 14)

- Will work on react-share and react-helmet libraries for introducing a share button.
- Will work on different post templates for EOS Icons, EOS Design System, and EOS User Story that will be shared on various platforms.
- Writing the blog about the work done.

Week 2 (June 14 - June 21)

- Will work on implementation of search stories feature for the home page.

- Improve the search functionality that will work for story description and numeral input also.
- Writing the blog about the work done.

Week 3 (June 21 - June 28)

- Will discuss with the mentors and work on implementation of request templates on the frontend.
- Writing the blog about the work done.

Week 4 (June 28 - July 5)

- Will continue with the previous week's work.
- Will also work on the display of priority labels that are selected while formation of new stories.
- Writing the blog about the work done.

Week 5 (July 5 - July 12)

- Will work on the backend and implement the functionality of making new stories with the help of request templates.
- Will work on writing tests.
- Writing the blog about the work done.

First Evaluation(July 12 - July 16)

Week 6 (July 12 - July 19)

- Continue with the previous week's work.
- Will implement the feature of sorting of stories on the basis of priority for home page, my stories page, and user profile page.
- Will discuss with the mentors about which of the email notifications functionalities that should be implemented and the preferred tech stack for implementation.
- Writing the blog about the work done.

Week 7 (July 19 - July 26)

- I will spend this week implementing email notification functionality for all of the proposed features discussed with mentors.
- Writing the blog about the work done.

Week 8 (July 26 - August 2)

- Continue with the previous week's work.

- Writing the blog about the work done.

Week 9 (August 2 - August 9)

- Will start working with the react-mentions library.
- Writing the blog about the work done.

Week 10 (August 9 - August 16)

- Will work on notification feature for mentioning functionality.
- Will work on writing unit tests.
- Writing the blog about the work done.

Week 11 (August 16 - August 23)

- Buffer week to do any remaining work.
- Finalize the code.

Final Evaluation(August 16 - August 23)

Previous Experience

Open Source Contributor: Contributed to project [Awesome Portfolio websites](#) and [Doc2Pen](#) during Contributor's Hack, 2020 organized by HackinCodes. I was the third top contributor for the project among almost 800 participants from all over the country.

[Link to Certificate](#)

[Link to contributions in Awesome Portfolio Websites](#)

[Link to contributions in Doc2Pen](#)

Web Development Intern: Worked as a web development intern as a part of the Graduate Rotational Internship Program organized by The Sparks Foundation. During the internship, I built a [credit management web application](#) through which credit points can be transferred from one person to another.

[Link to Certificate](#)

Participant: I actively participated in DS-Algo and C/C++ teams during The Uplift Project, 2020 organized by GirlScript Foundation, and worked on the implementation of OpenCV in C++.

[Link to certificate for DS-Algo Team](#)

[Link to certificate for C/C++ Team](#)

Open Source Contributor: I was an open-source contributor for the project Algo-DS Notes during GirlScript Summer of Code, 2020.

[Link to Certificate](#)

Personal Projects

Food Finder: A food ordering web application built using PHP, MySQL, JavaScript, HTML, and CSS where a user can sign up and order food from given different restaurants, choose food from the given menu, and can also give their reviews and ratings.

[Github Repository Link](#)

[Project Link](#)

Chat-Room: A real-time chat application built using ReactJs, NodeJs, and Socket.io that allows users to join a common chat room and chat.

[Github Repository Link](#)

[Project Link](#)

Tic-Tac-Toe: A tic-tac-toe single-player game built using ReactJs.

[Github Repository Link](#)

[Project Link](#)

To-Do List: A web application built using PHP, MySQL, HTML, and CSS where users can sign up and maintain their own to-do list by adding a task, deleting a task, and marking a task as done.

[Github Repository Link](#)

[Project Link](#)

The Foodiez: Frontend of a responsive website.

[Github Repository Link](#)

[Project Link](#)

About Me

I am a second-year undergraduate pursuing my bachelors in Computer Science and Engineering from Guru Jambheshwar University of Science and Technology, Hisar, Haryana, India. I am an open-source enthusiast and a keen learner. I love to solve real-world problems through code. Besides coding, I am indulged in reading also. I am an avid reader and you will always find me reading something in my free time.

What I liked the most about EOS was that the EOS community and all the mentors are really supportive. I have been contributing to EOS for quite some time now and I am familiar with the codebase. I am confident that I will be able to complete the project efficiently.

Other Commitments During Summer

I will not be participating in any other internships during the GSOC program. I would be able to devote 18-20 hours a week and more if required.

Looking forward to working with you,

Harshita Mangla