Disclaimer

- Due to limitation of the export functionality on Atlassian Confluence, some sections of the PDF might not be rendered well.
- Further, we have spent a considerable amount of effort to not duplicate information across a number of our documentation sites - Jira, Confluence and Github and in the larger endeavor of information consistency, we hope to keep using Atlassian provided integration for tools like Jira and GitHub.
- Link to Jira: https://comp90082-2024-si-koala.atlassian.net/jira/software/projects/SI/boards/1
- Link to Confluence: https://comp90082-2024-si-koala.atlassian.net/wiki/spaces/comp900822/overview
- Link to GitHub: https://github.com/COMP90082-2024-SM1/SI-Koala

comp90082-2024-si-koala

Welcome to The Confluence!

The following document contains all the information about the project Studies in Language Assessment (SiLA).

You can easily navigate to any of the sections from the following:

Project Information - SiLA

Executive Summary 🚀

Studies in Language Assessment (SiLA) is an official peer-reviewed publication journal of the Association for Language Test and Assessment of New Zealand(ALTAANZ) who are looking to use technology to enable its users to better share their work and get feedback through a **Manuscript Submission Platform**. In collaboration with students at The University of Melbourne, SiLA aims to begin their journey for better technology enablement for its users. During the phase of the next 8-9 weeks, students will work closely with stakeholders to deliver on key requirements stated by SiLA.

Background Information.

The current operational framework of Studies in Language Assessment (SiLA) involves a meticulous peer review process overseen by the Association for Language Testing and Assessment of Australia and New Zealand (ALTAANZ). Manuscripts are submitted via email to sila.editors@gmail.com, where they undergo an initial check by the editorial assistant before review by co-editors and external scholars. The review is double-blind, ensuring impartiality and transparency. Accepted manuscripts are edited, formatted, and published with a DOI, following ethical and academic standards.

Users, including authors, editors, and external reviewers, must adhere to responsibilities such as originality, ethical conduct, confidentiality, and acknowledgment of contributions. Editors play a crucial role in maintaining fairness and integrity throughout the review process. External reviewers are expected to provide impartial feedback within a specified timeframe and report conflicts of interest. The journal is also developing guidelines to address potential malpractices like plagiarism or ethical violations, ensuring the publication's integrity and credibility.

The operational structure of SiLA, an esteemed international journal on language testing and assessment, is well-defined and meticulous. Managed by ALTAANZ, the journal maintains a robust peer review process, ensuring transparency and adherence to ethical standards. Authors submit manuscripts via email, which undergo a rigorous double-blind review by expert scholars. Accepted papers are meticulously edited, formatted, and published with a DOI, reflecting SiLA's commitment to scholarly excellence.

Authors, editors, and external reviewers play crucial roles in upholding the journal's integrity. Authors must ensure originality, ethical conduct, and proper acknowledgment of contributions. Editors oversee the review process, maintaining fairness and confidentiality. External reviewers provide impartial feedback and report conflicts of interest, contributing to the rigorous evaluation of submissions.

You can Read more about how the process is being conducted in the document below:

SiLA publication policies draft 27-02-24.docx

The following is an overview of how content is organized in this space (comp90082-2024-si-koala).

- Team overview available at Meet The SI-Koala Team.
- The Information About the Project is Available at Project Information - SiLA
- All meeting notes are available in Meetings
- The Project Official Documentation is available in Documentation
- Project Requirements are Highlighted in Key Requirements
- Some quick start templates are available in Templates.

FAQ

- Useful links:
  - GitHub Teams
  - Miro Board

- SiLA

👋 Meet The SI-Koala Team

- Akash Renuka Ashok
- Master of IT - AI

- Edwin Zhu
- Master of IT - AI

- Jai Phookan
- Master of IT - Distributed Computing

- Rishabh Srivastava (***"Rish"***)
- Master of IT - AI

- Saurabh Zingade
- Master of IT - AI

- Vijay Narayan Venkatesh
- Master of IT - AI

Lin Li

Contact Info

- akashrenukaa@student.unimelb.edu.au
- xiyanedwinz@student.unimelb.edu.au
- jphookan@student.unimelb.edu.au
- rssriv@student.unimelb.edu.au
- szingade@student.unimelb.edu.au
- vijaynarayan@student.unimelb.edu.au
- ll8@unimelb.edu.au (Mentor)

🔑 Key Requirements

- Requirements and User Stories + Non-Scope
- Prototypes
- User Personas
- Non-Functional Requirements
- Human-Centered Design

Requirements and User Stories + Non-Scope

2 Story Points = 1 Day's Work Per Person

Key Requirements

Type	Key	Summary	Status	Priority	Assignee	Co-Assignee	Story point...
	SI-13	User Registration and Management	IN PROGRESS	=	Saurabh Zingade	Akash Renuka Ashok	87
	SI-14	Manuscript Submission	IN PROGRESS	=	Jai Phookan	Vijay Venkatesh	35
	SI-15	Manuscript Tracking and Management	TO DO	=	Akash Renuka Ashok	Saurabh Zingade	50
	SI-16	Peer Review System	TO DO	=	Akash Renuka Ashok	xiyanedwinz	13
	SI-17	Editorial Decision-Making Tools	TO DO	=	Vijay Venkatesh	Unassigned	16.5
	SI-18	Data Security and Confidentiality	TO DO	=	Rishabh Srivastava	xiyanedwinz	23
	SI-19	Scalability and Flexibility	TO DO	=	xiyanedwinz	Rishabh Srivastava	12
	SI-20	Analytics and Reporting	TO DO	=	xiyanedwinz	Unassigned	8

8 items

Synced just now

User Stories

Type	Key	Summary
	SI-21	As a user, I want to be able to securely login to the platform so that I can access the system
	SI-22	As an author, I want to be able to signup on the platform so that I can login securely
	SI-23	As an editor/editor-assistant, I want to create accounts on behalf of editors and reviewers so that they can login t
	SI-24	As a user, I want to be able to reset my credentials so that I can keep my account secure
	SI-25	As a user, I want to be able to edit my account details so that I have latest information available
	SI-26	As a user, I want to be able to switch between assigned roles so that I am aware of my actions and responsibiliti
	SI-27	As an editor/editor-assistant, I should be able to assign roles to users so that there is a central management of r
	SI-28	As an author, I want to be able to submit a manuscript so that it can be considered for publishing
	SI-29	As an author, I want to be able to receive an email confirmation when the submission is made so that I am awar
	SI-30	As an author, I want a simple and easy to use UI so that I am aware of the steps in the manuscript submission
	SI-31	As an editor, I want to see progress of submissions so that I can track progress

Type	Key	Summary

23 items

Synced just now

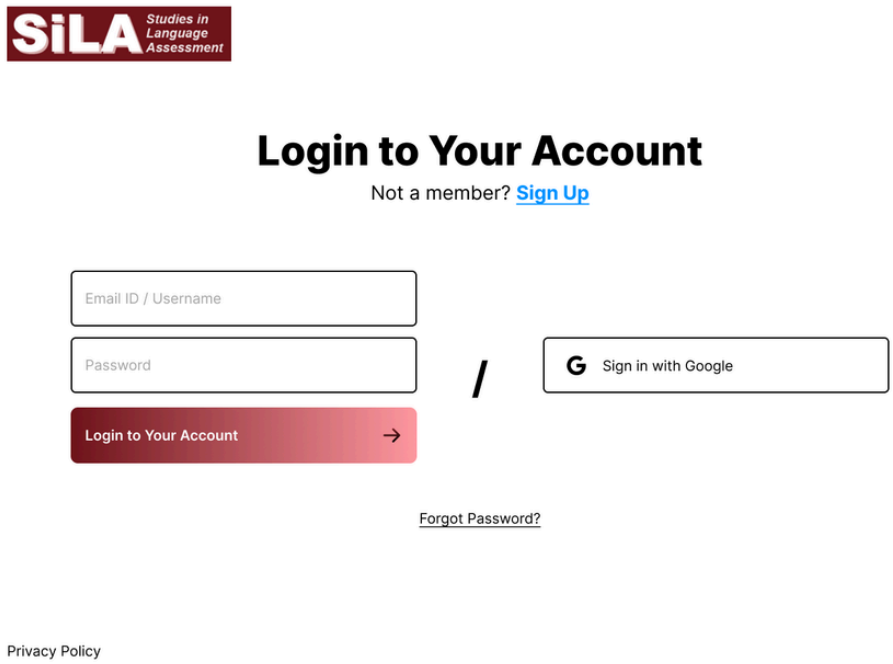
Non-Scope

- **Real-time collaboration tools**: The platform will not support real-time manuscript editing features akin to Google Docs, due to the complex technical requirements involved
- **Mobile app development**: Dedicated mobile applications for iOS and Android to provide full access to the platform's features will significantly expand the project's scope and will not be pursued
- **Screen orientation:** Portrait screens will not be considered for development. The platform will only support development in landscape mode in browsers on both mobile and desktop devices
- **Advanced analytics and reporting tools:** Advanced analytics and reporting capabilities, such as predictive analytics, machine learning models for pattern recognition etc will not be implemented
- **Advanced scalability:** Advanced scalability techniques, such as dynamic resource allocation via cloud services (e.g., AWS Auto Scaling, Kubernetes), are excluded from the initial phase
- **Advanced communication capabilities:** The communication module will offer basic text-based interactions only, excluding real-time messaging, voice, or video conferencing capabilities to keep the project scope manageable.
- **Handling different issue types:** Dealing with different issue types at the journal (i.e. Special Issue, Regular Issue, etc) will not be pursued

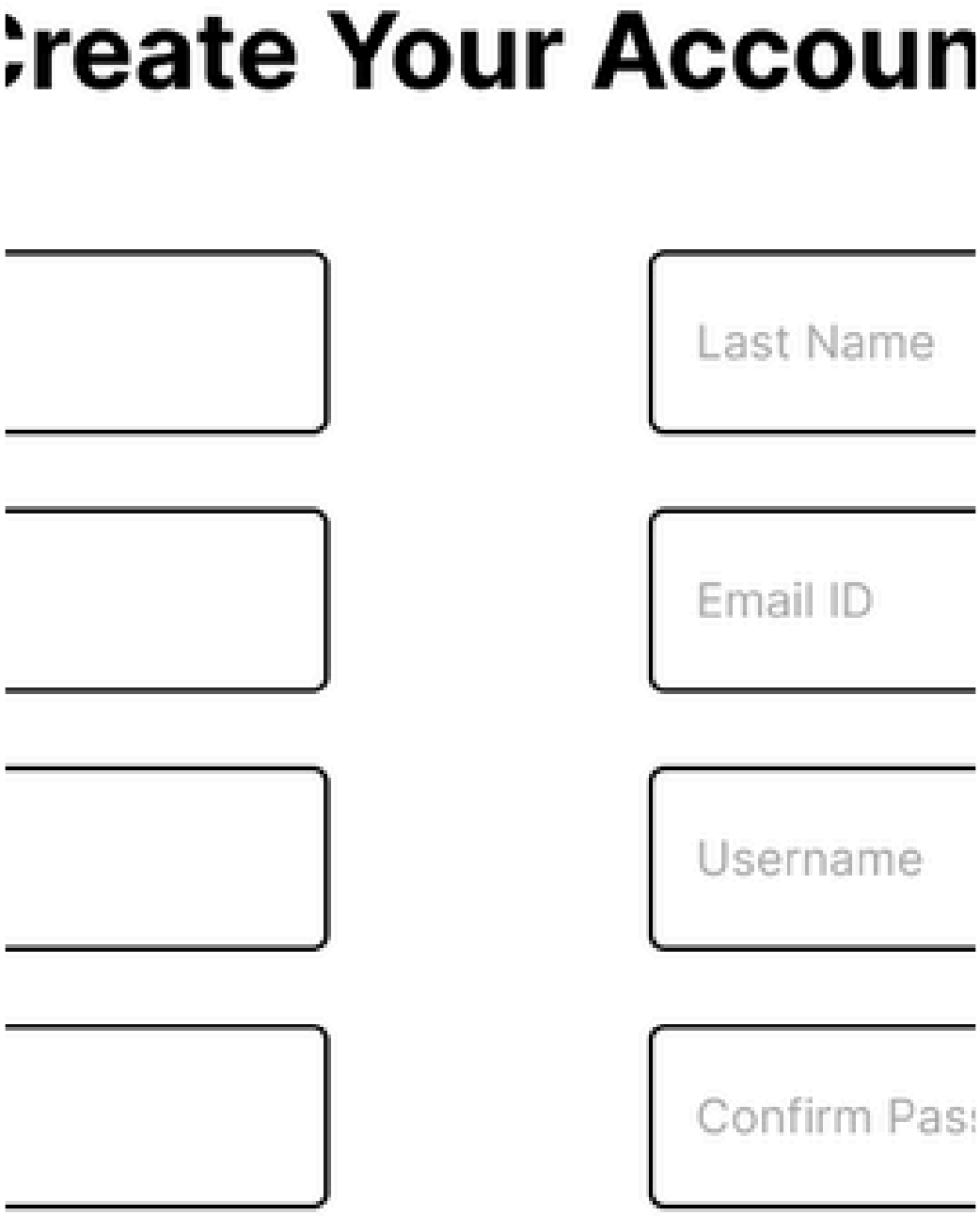
Prototypes

The following Section contains the Protype UI we will develop during this project.
This will be revised after discussion with the Client.

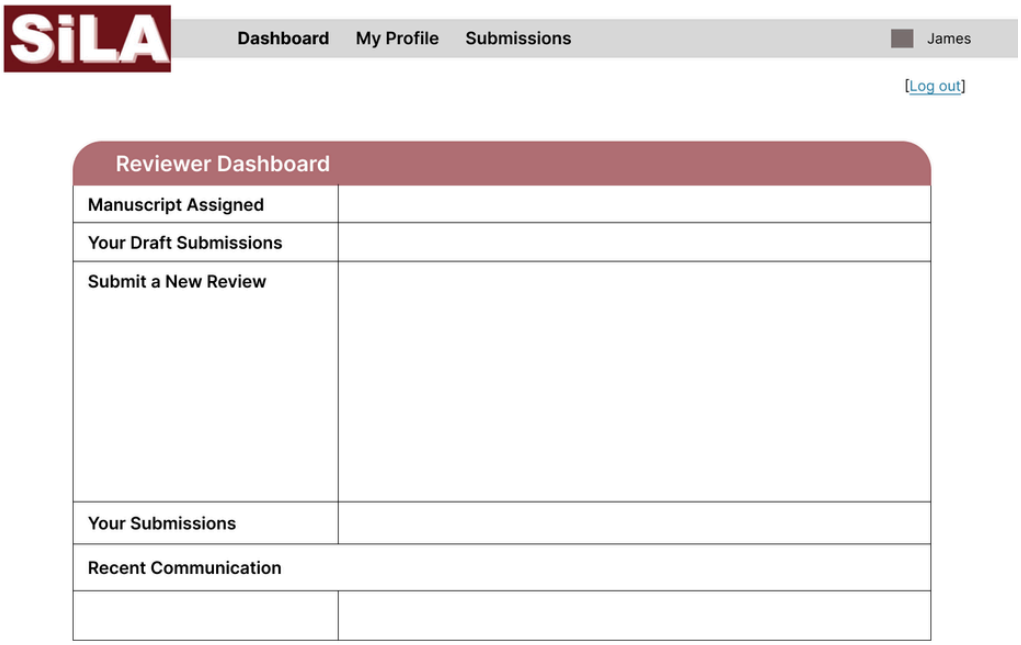
Login Page:

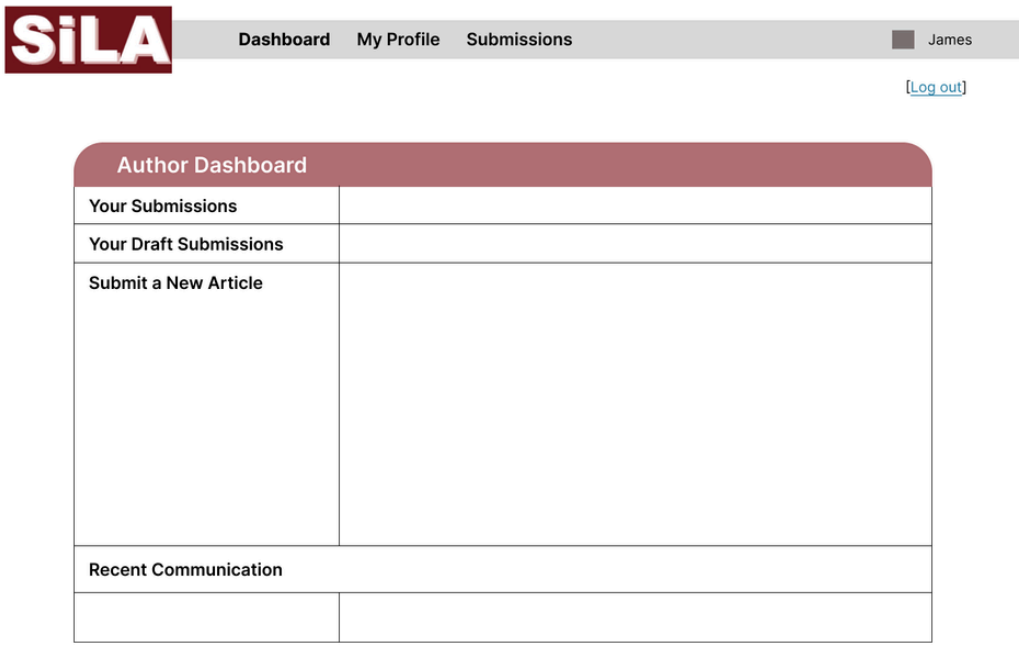


Sign-Up Page:



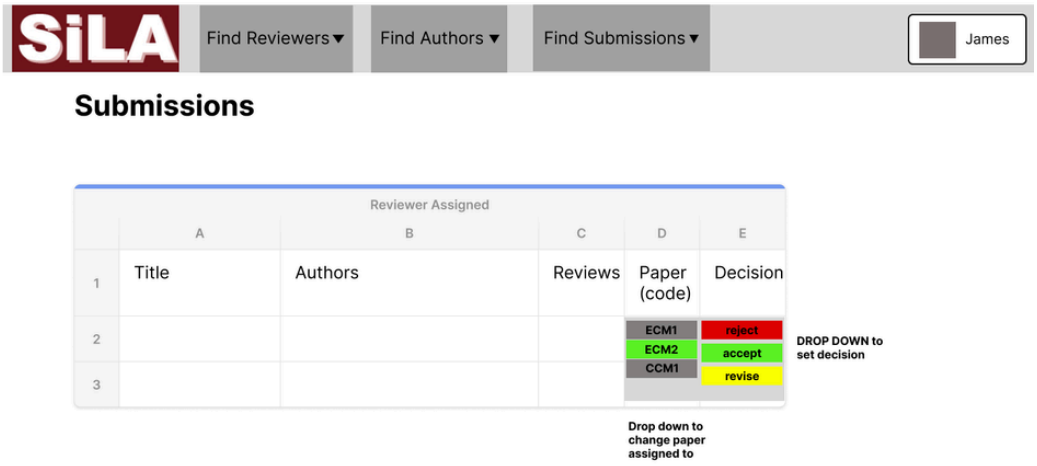
Home Page Dashboard



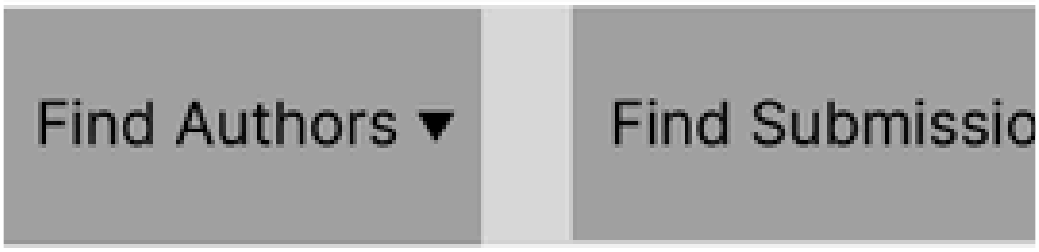


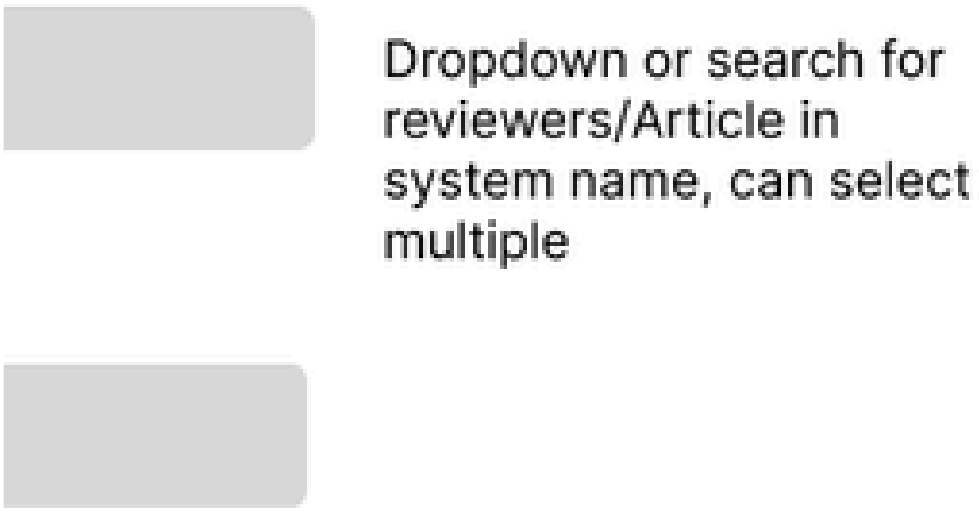
Editors View

Submissions Dashboard:

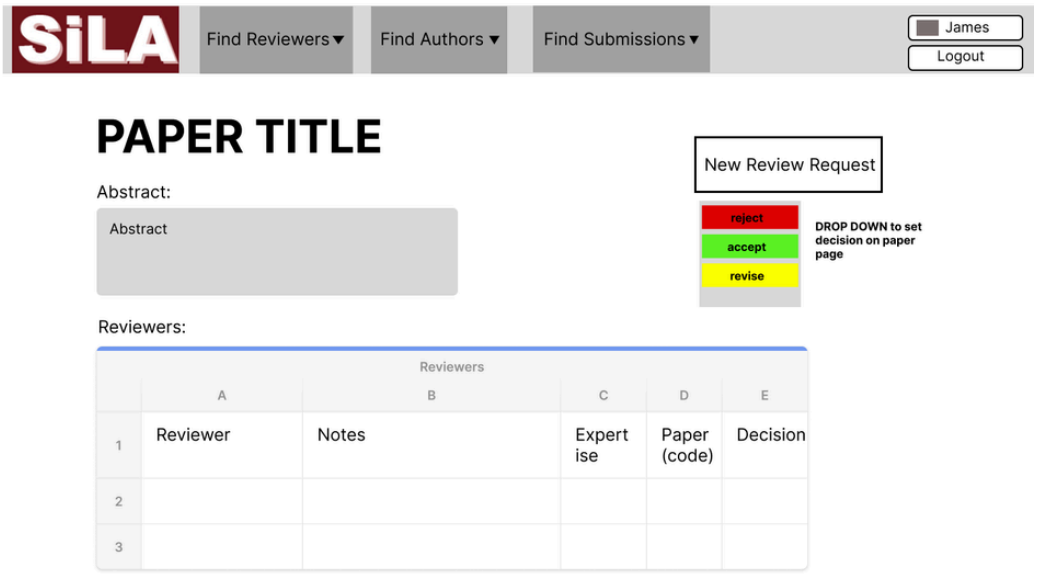


Assign Reviewer(s) to a paper (or review request):





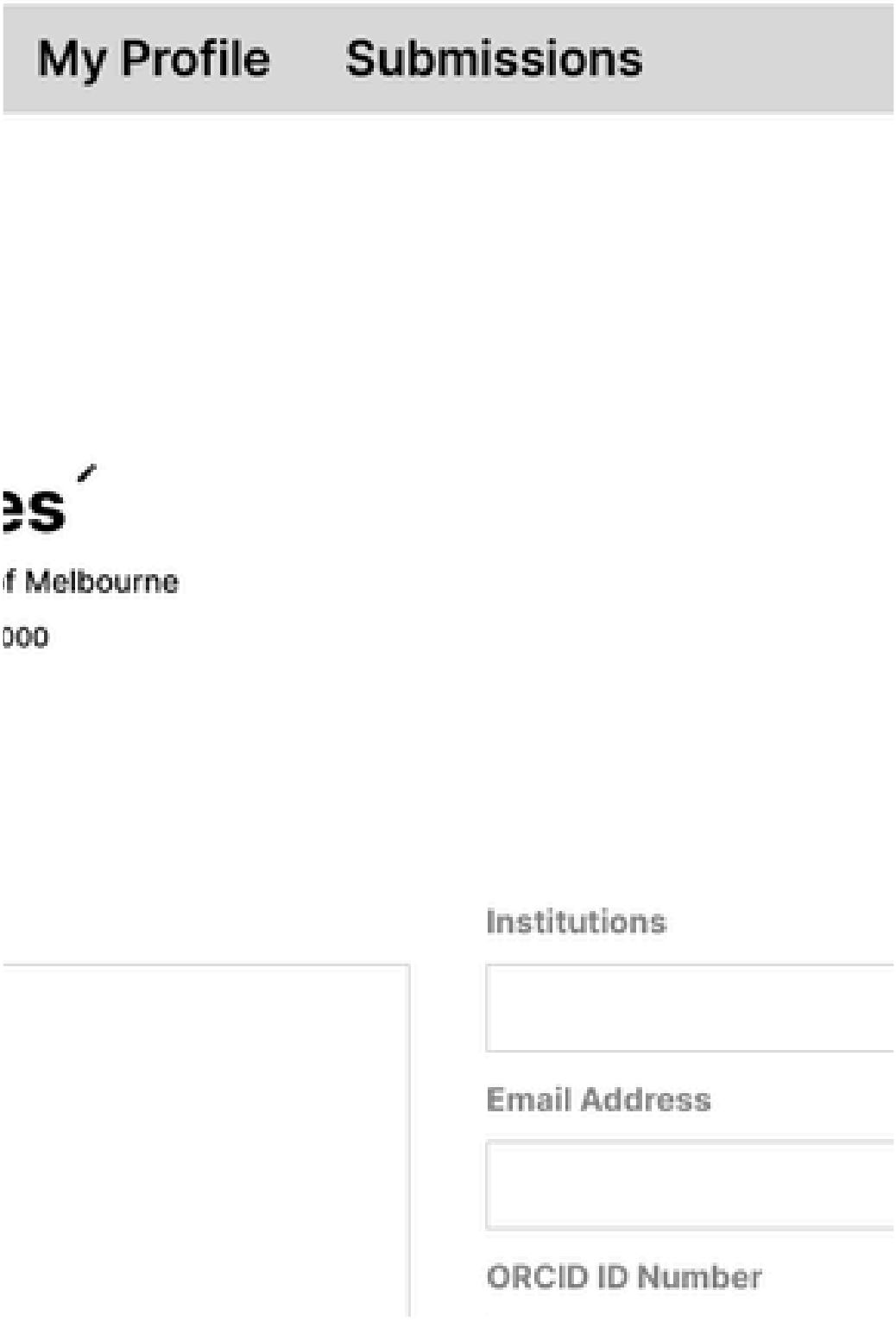
Individual Paper Page:



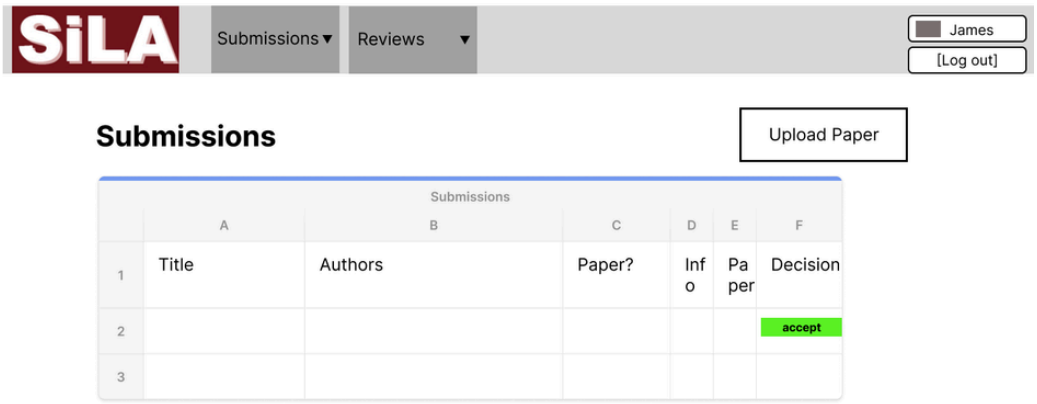
Authors/Reviewers View: **(can only see each tab if assigned to the respective role)**

Home Page

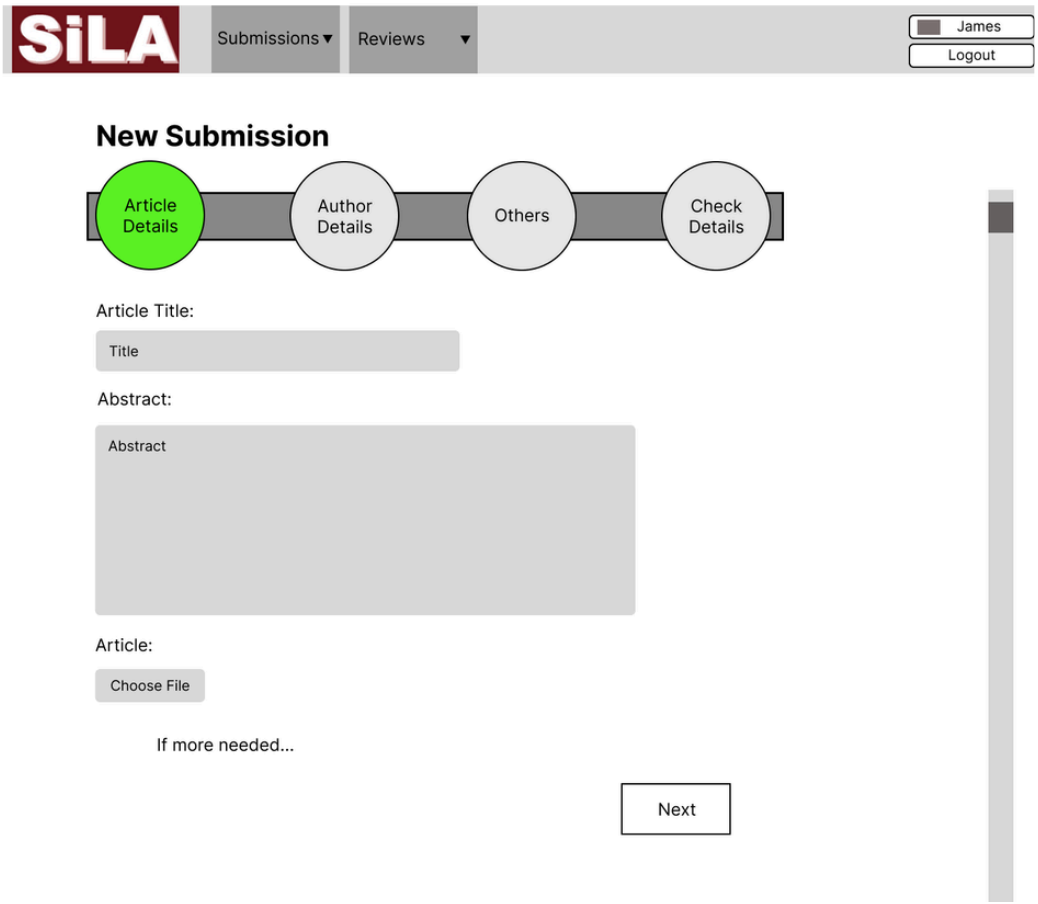
Profile Page:



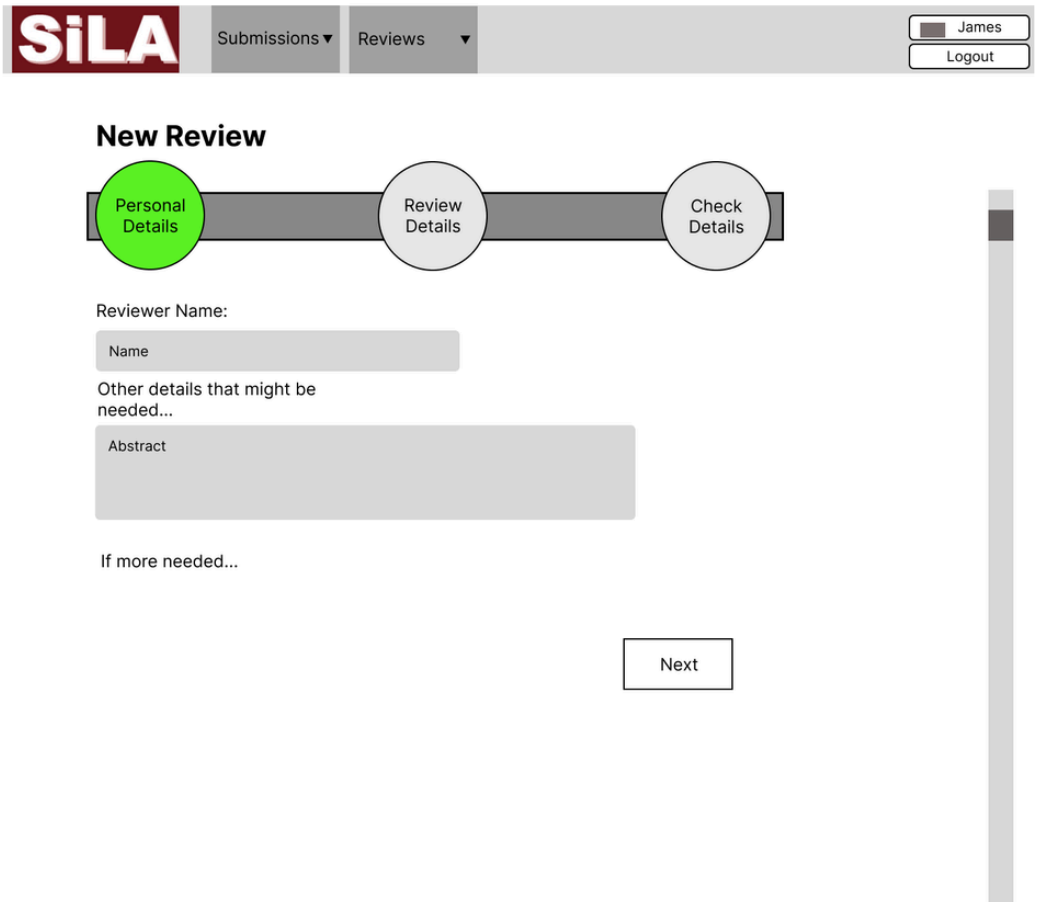
Submissions Dashboard:



Submissions upload Tracker:



Reviewer To check for a new Review:



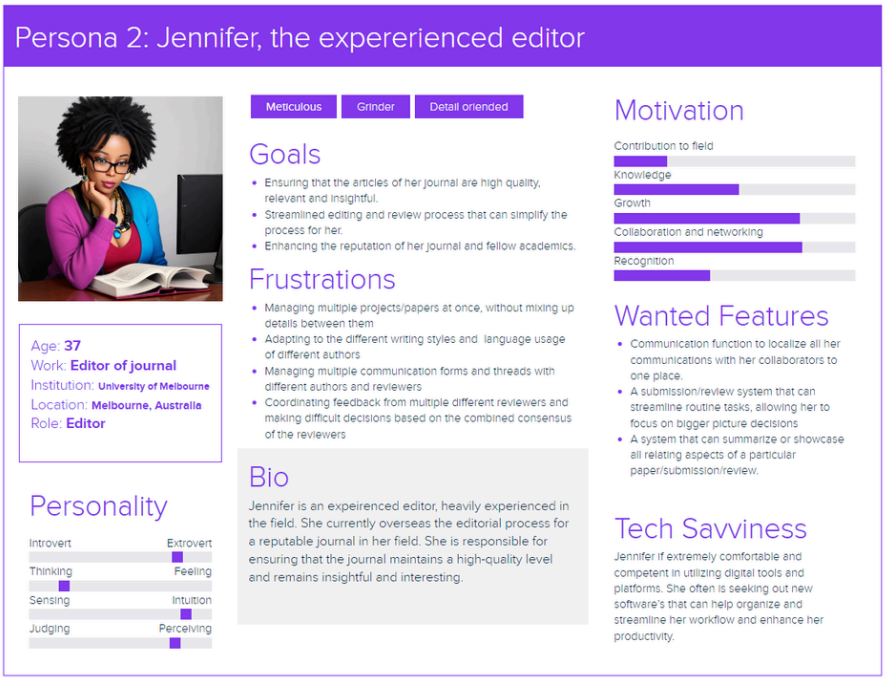
User Personas

The personas have been identified after consultation with the client on who the users of the platform might be and what skill set they might possess. Personas have been designed adopting similarities to our clients with respect to the internal users of the platform. For external users like authors and reviewers, our choice has been made after careful consideration of the client feedback about their current audience and future prospects.

The following main personas have been identified in the project.

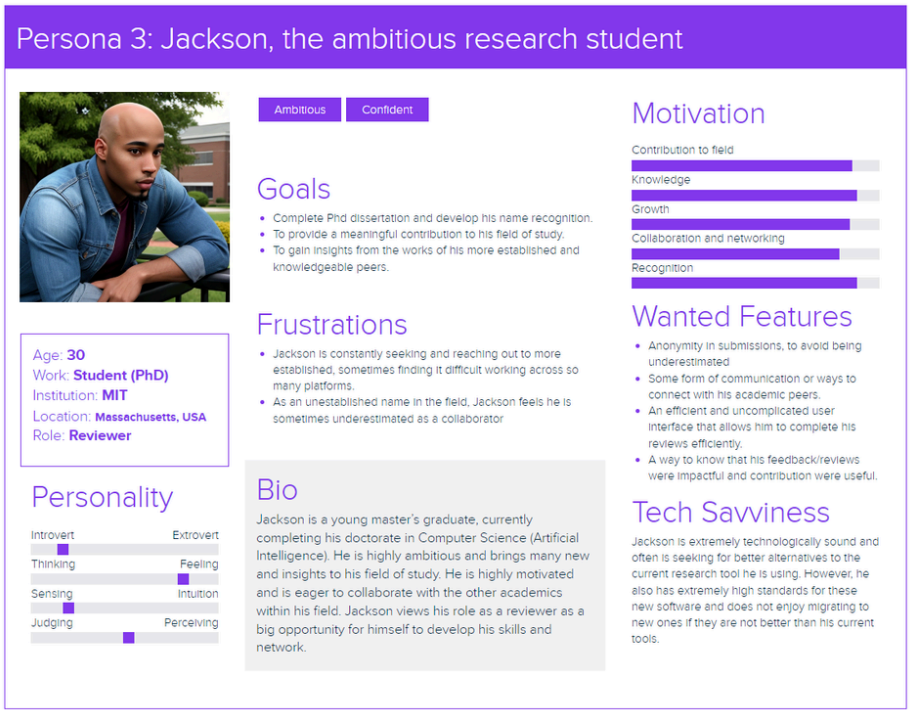
Persona 2: Jennifer, the experienced editor

Personal Details	
Name	Jennifer
Age	37
Occupation	Editor of a reputable journal
Role	Editor
Institution	University of Melbourne
Location	Melbourne, Australia



Persona 3: Jackson, the ambitious research student

Personal Details	
Name	Jackson
Age	30
Occupation	Student (completeing PhD)
Role	Reviewer
Institution	MIT
Location	Massachusetts, USA



Personal Details	
Name	Alex
Age	28
Occupation	University Tutor
Role	Editorial Assistant
Institution	University of Queensland
Location	Queensland, Australia

💡 Non-Functional Requirements

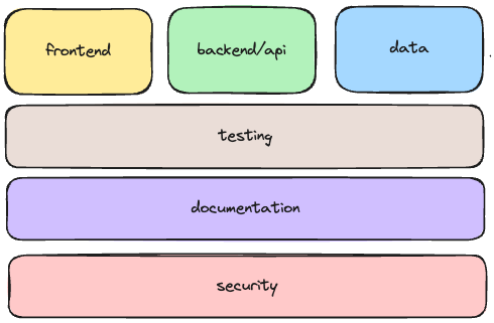
Overview

This document states the non-functional decisions taken to achieve the key user requirements. We have taken a holistic approach in picking the right tools for the job which make sure that we as students are learning enough during development, the solution is easily manageable by the client post handover, the operating cost of the solution is kept as little as possible, and the solution adheres to industry best practices.

This document talks about what we think is the right tool in the context of the project and it ***DOES NOT*** talk about why we did not use an alternative tools.

Non Functional Domain

We at team Koala, believe in a shared responsibility and domain ownership model where various domain owners collaborate with developer to collectively work together for the delivery of a feature to the client.



- ***Top Domain***
  - **Frontend** - Everything related to the user experience.
  - **Backend/API** - Everything related to connecting the frontend with the data
  - **Data** - Everything related to the data of the application
- ***Shared Domains (everyone's responsibility)***
  - **Testing** - Automated test cases
  - **Documentation** - Document-As-You-Go
  - **Security** - Making sure every feature is secure

Responsibility Structure

While everyone will be working across all the domains for delivery and learning experience in general, we decided it would be best to have domain owners who spearhead in the field through prior experience. They will be making key decision on what choices to make and architectural decisions and help in setting up a better learning experience for the rest of the team. The role of a **supporter** is to work closely with the owners to make sure that in later stages, domain owners are not bottle necks in the review process and they can provide more bandwidth with peer code review process.

Domain	Owner	Support
Frontend	@Akash Renuka Ashok	@Jai Phookan @Saurabh Zingade

Backend/API	@Saurabh Zingade @Vijay Venkatesh	@xiyanedwinz @Jai Phookan
Data	@Rishabh Srivastava	@xiyanedwinz

** ***Backend/API*** is expected to be a rather larger domain.

Technology Stack

Frontend

- ***Tool*** - React
- ***Reason***
  - Enough quality experience in the team
  - Robust framework that can be later extended to Android and iOS platforms
  - Ease of use
  - Industry standard

Backend/API

- ***Tool*** - Node.js
- ***Reason***
  - Enough quality experience in the team
  - Abundant availability of support resources and packages
  - Ease of use
  - Industry standard

Data

- ***Tool*** - MySQL db
- ***Reason***
  - Enough quality experience in the team
  - Data models are mostly relational and repeated. Hence, MySQL will be most efficient
  - Absence of graph based or heterogeneous object structures that call for NoSQL databases
  - Ease of use
  - Industry standard

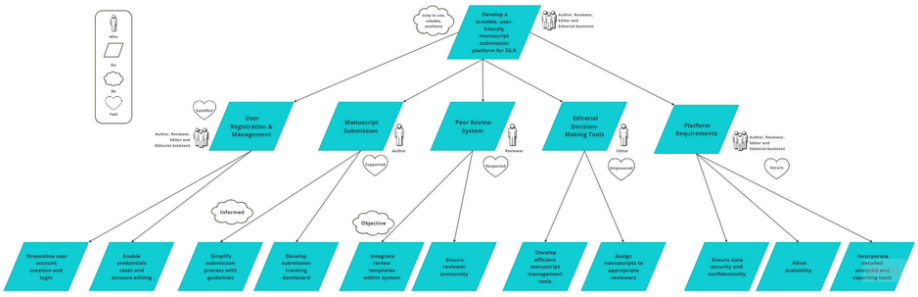
Thought Leadership

As a team, we are keen to do ***DevOps*** and live deployment on a Cloud Provider. However, we are parking that decision for later sprints due to limitation for a number of reasons.

Use a CI server like GitHub Actions/CircleCI, etc to deploy the application to Amazon Web Service. Everything in free tier

Motivational Model

The following contains information about the goals that would be accomplished throughout the project.



DO-BE-FEEL List

Roles 👤	Do 📝	Be 💡	Feel ❤️
Author	Submit manuscripts, revise submissions, check status.	Informed, engaged, supported, easy to use.	Confident, valued, satisfied with feedback.
Reviewer	Receive manuscripts, review manuscripts, submit reviews.	Objective, thorough, timely.	Respected, engaged, appreciated.
Editor	Make editorial decisions, communicate with authors/reviewers.	Decisive, insightful, communicative.	Empowered, responsible, proud of the journal's standards and integrity.
Editorial Assistant	Assist in manuscript management, facilitate communication.	Organized, supportive, proactive.	Valued for their role, efficient, integral to the process.

🌈 Unconscious Bias & Inclusivity

- Can users add their desired pronouns?
- Does the system provide any support for users with special needs like text-to-voice, voice-to-text, etc?

*This Section will be constantly updated throughout the project.

📑 Documentation

This Section contains all the relevant documentation for the project.
This section will be updated as we progress through the project.

- Live Deployment Link!
- Code Reviews.
- Quality Assurance
- SDLC Model
- SiLA ERD
- Ethics Centred Development
- Security

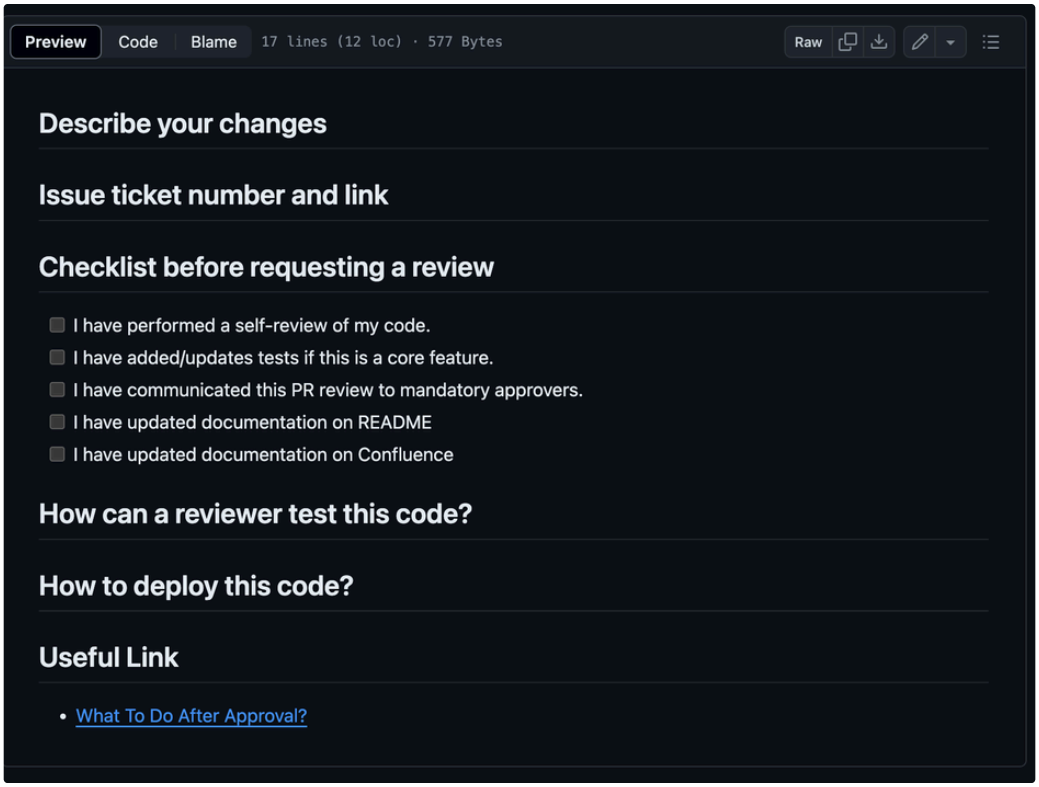
🤖 Code Reviews.

Manual Code Review

All code changes mostly fall into `/backend`, `/frontend` or `/data` which impact the application. All code changes come via PR and requires a mandatory approval from the below list of people.

Domain	Owner	Support
Frontend	@Akash Renuka Ashok	@Jai Phookan @Saurabh Zingade
Backend/API	@Saurabh Zingade @Vijay Venkatesh	@xiyanedwinz @Jai Phookan
Data	@Rishabh Srivastava	@xiyanedwinz

All of the PR's follow a template like below:



ref: https://github.com/COMP90082-2024-SM1/SI-Koala/blob/main/.github/pull_request_template.md

Example PR: https://github.com/COMP90082-2024-SM1/SI-Koala/pull/106 RESTRICTED CONTENT

- Reviews are requested from those people who have best insights on the impact of the incoming change and further code review process involves conversations on the PR, on slack or completely offline if the team is working together in person.

Automated Code Review

- We are using `agogear/chatgpt-pr-review@0.0.1` Github Actions plugin to review incoming changes to the `main` branch.
  - ref: https://github.com/COMP90082-2024-SM1/SI-Koala/blob/main/.github/workflows/chatgpt_code_review.yml
- OpenAI suggested changes are reviewed by the team internally to cherry pick what reviews should be actioned. (eg. The chatGPT review gave us the feedback to add comprehensive documentation and we will be adding that during the next Sprint)
- Example PR:
  - https://github.com/COMP90082-2024-SM1/SI-Koala/pull/121
  - https://github.com/COMP90082-2024-SM1/SI-Koala/pull/122

🚨 Quality Assurance

We have developed automated test cases for the APIs, so that we are aware that they work as expected.
There are two types of test cases that have been developed as of now.
1.) Unit Test cases.(UTC)
2.) Integration Test cases. (ITC)

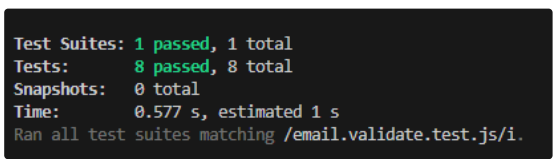
⭐ Unit Test Cases.

The Unit test cases are for the smaller APIs that are used through the application.

1.) Email Validation

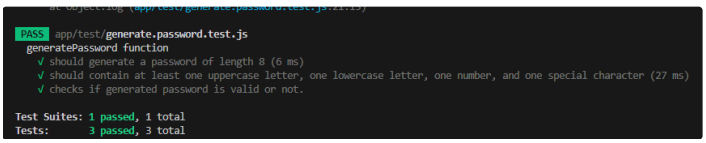
TEST SUITE ID	**UTCS1**
TEST CASE ID	**UTC1.1 :** Check if the email is valid.
Test case Summary	Verify the functionality of valid email functions for different email formats.
Prerequisites	The relevant functions are accessible and the path is proper. The database should be connected and running on Linux or WSL Portal. Also ,all the relevant packages are installed. You can easily do this by the command `npm i`
Test Script/Procedure	Run the command `npm test email.validate.test.js` from the test folder where all the test cases are implemented.

Test Data	`Test Data Test emails:`
	`- 'test'`
	`- 'test@gmail.com'`
	`- 'testgamil.com'`
	`- 'mask@yahoo.com'`
	`- 'Asciiii@'`
	`- 'ronaldo@alnassar.com'`
	`- 'bruno@manunited'`
	`- 'bestiii123#aaa'`

Expected Result	`[false,true,false,true,false,true,false,false]`
Actual Result	`[false,true,false,true,false,true,false,false]` 
Remarks	Multiple emails can be added and checked in this unit test case.
Status	**PASS**

Created By	@Saurabh Zingade
Executed by	@Saurabh Zingade
Latest Date of Execution	May 2, 2024

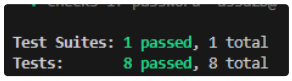
2.) Generate Password

TEST SUITE ID	**UTCS2**
TEST CASE ID	**UTC2.1 :** Should generate a password of length 8. **UTC2.2 :** Should contain at least one uppercase letter, one lowercase letter, one number, and one special character. **UTC2.3** : Check if the generated email is valid or not.
Test case Summary	Verify the functionality of generating one time passwords for the users.
Prerequisites	The relevant functions are accessible and the path is proper. The database should be connected and running on Linux or WSL Portal. Also,all the relevant packages are installed. You can easily do this by the command `npm i`
Test Script/Procedure	Run the command `npm test generate.password.test.js` from the test folder where all the test cases are implemented.
Test Data	`None (dynamic generation of passwords).`
Expected Result	The generate password function will generate the password that should be of 8 characters and should contain at least one uppercase letter,one lowercase letter, one number and one special character. It will also check again if the generated password is valid or not.
Actual Result	The password generated was `` `Zw1%GfhP` `` which satisfies the condition mentioned above. 
STATUS	**PASS**
Remarks	Everytime a password is generated is it generated randomly which satisfies the condition.
Created By	@xiyanedwinz
Executed by	@xiyanedwinz
Latest Date of Execution	May 2, 2024

3.) Password Check.

TEST SUITE ID	UTCS3
TEST CASE ID	**UTC3.1 :** Validate the given password.
Test case Summary	Verify the functionality of password validation.
Prerequisites	The relevant functions are accessible and the path is proper. The database should be connected and running on Linux or WSL Portal. Also,all the relevant packages are installed. You can easily do this by the command `npm i`
Test Script/Procedure	Run the command `npm test password.policy.test.js` from the test folder where all the test cases are implemented.

Test Data	`Test Data Test passwords:`
	`- 'testing'`
	`- 'testing124'`
	`- 'hello'`
	`- 'Hdj@99sDD!'`
	`- 'maskdoo222'`
	`- 'okd22@@ddd'`
	`- 'dsasddjjiD@'`
	`- 'assd2D@'`

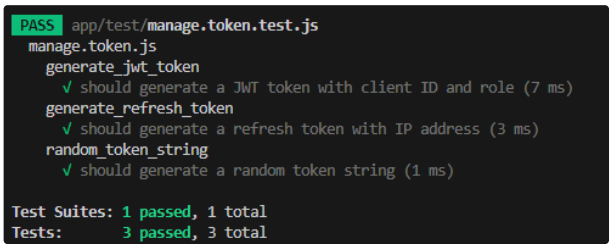
Expected Result	`[false,false,false,true,false,false,false,false]`
Actual Result	`[false,false,false,true,false,false,false,false]` The function clearly identifies both the valid and invalid passwords in the provided test cases. 
STATUS	**PASS**
Remarks	This is used to check whether the password is valid or not .
Created By	@Vijay Venkatesh
Executed by	@Vijay Venkatesh
Latest Date of Execution	May 2, 2024

4.) Manage Token

TEST SUITE ID	UTCS4
TEST CASE ID	**UTC4.1 :** Generate JWT Token **UTC4.2 :** Generate Refresh Token **UTC4.3 :** Random Token String.
Test case Summary	Verify the functionality of the manage token module which is used for token generation.

Prerequisites	The relevant functions are accessible and the path is proper. The database should be connected and running on Linux or WSL Portal. Also, all the relevant packages are installed. You can easily do this by the command `npm i`
Test Script/Procedure	Run the command `npm author.submission.test.js` from the test folder where all the test cases are implemented.

Test Data	`Mock client ID: 'mockClientId'`
	`Mock client role: 'mockClientRole'`
	`Mock IP address: 'mockIPAddress'`

Expected Result	• The module will generate tokens with the correct properties and lengths. • It checks Checks if `jwt.sign` is called with the correct parameters, including the client ID and role • Checks if the generated token contains a `refresh_token` property with a length of 80 characters (since it's generated in hex). • Verifies that the generated token contains an `ip_address` property matching the provided IP address. • Verifies that the generated random token string has a length of 80 characters
Actual Result	All the checks were done successfully in the test case. 
STATUS	**PASS**
Remarks	The test cases ensure that token generation functions in manage.token.js behave as intended, providing essential security features for authentication and authorization.
Created By	@Rishabh Srivastava
Executed by	@Rishabh Srivastava
Latest Date of Execution	May 2, 2024

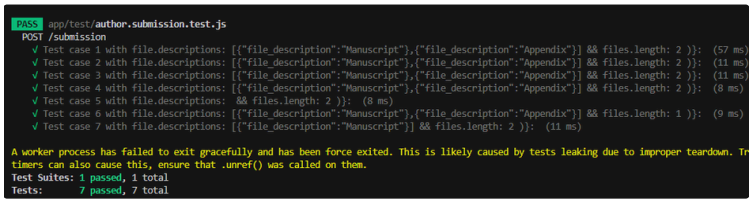
🌟 Integration Test Cases.

The Integration test cases are combination of multiple APIs that need to be synced in order to work in the expected way and provide the expected result.

1.) Manuscript Submission by Author

TEST SUITE ID	**ITCS1**

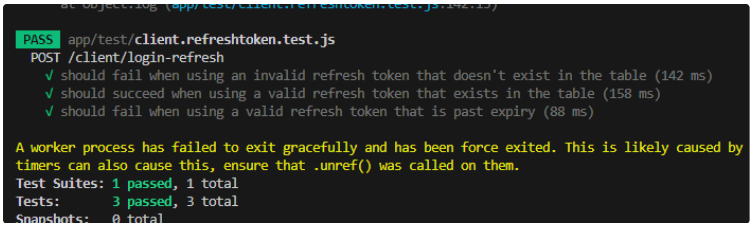
TEST CASE ID	**ITC1.1:** Correct Submission: Number of Files Submitted & File Descriptions are equal. **ITC1.2:** Correct Submission Object + Valid Filetype Submissions. **ITC1.3:** Correct Submission Object + 1 Invalid Filetype Submissions + 1 Valid Filetype. **ITC1.4:** Correct Submission Object + 2 Invalid Filetype Submissions. **ITC1.5:** Empty File Description **ITC1.6:** The Number of file_descriptions (1) does not equal the number of files submitted (2). **ITC1.7:**Number of file_descriptions (2) does not equal number of files submitted (1).
Test case Summary	Verify the functionality of the /submission endpoints for handling the author Submission of the manuscripts.
Prerequisites	The relevant functions are accessible and the path is proper. The database should be connected and running on Linux or WSL Portal. Also, all the relevant packages are installed. You can easily do this by the command `npm i`
Test Script/Procedure	Run the command `npm author.submission.test.js` from the test folder where all the test cases are implemented.
Test Data	``` - submission_title: "My Awesome Research Paper" - submission_type: "Normal" - abstract: "This is a summary of my awesome research paper" - acknowledgements: "None" - conflict_of_interest: "None" - authors: "John Doe, Jane Doe" - file_descriptions: JSON.stringify([{ file_description: "Manusc - filesArr: [{ name: "file1.docx", content: Buffer.from("Test file content 1" { name: "file2.pdf", content: Buffer.from("Test file content 2")] ``` This is the format of the data used for testing. Multiple types of data are tested. They include - Correct Submission Object + Valid Filetype Submissions - Correct Submission Object + 1 Invalid Filetype Submissions + 1 Valid Filetype - Correct Submission Object + 2 Invalid Filetype Submissions - Empty file_descriptions - Number of file_descriptions (1) does not equal number of files submitted (2) - Number of file_descriptions (2) does not equal number of files submitted (1)
Expected Result	1. **For Correct Submission: Number of Files Submitted & File Descriptions are equal:** ◦ Expected Result: The submission process should be successful, and the server should respond with a status code of 200 and a message indicating that the submission was successful. 2. **For Correct Submission Object + Valid Filetype Submissions:** ◦ Expected Result: The submission process should be successful for valid file types, and the server should respond with a status code of 200 and a message indicating that the submission was successful. 3. **For Correct Submission Object + 1 Invalid Filetype Submissions + 1 Valid Filetype:**

	◦ Expected Result: The submission process should fail due to the presence of invalid file types, and the server should respond with a status code indicating a client-side error (500 status code) and an error message indicating that only specific file types are allowed. 4. **For other scenarios (such as mismatched file descriptions or empty file descriptions):** ◦ Expected Result: The submission process should fail due to the mismatch between file descriptions and files or due to empty file descriptions, and the server should respond with a status code 500 indicating a client-side error and an error message indicating the reason for the failure.
Actual Result	All the checks were done successfully in the test case. 
STATUS	**PASS**
Remarks	This is an integration test case that tests the interaction of multiple components (server, database, file handling) to ensure the correct behavior of the /submission endpoint under various scenarios.
Created By	@Jai Phookan @Vijay Venkatesh
Executed by	@Jai Phookan @Vijay Venkatesh
Latest Date of Execution	May 2, 2024

2.) Client login with Refresh Token.

TEST SUITE ID	**ITCS2**
TEST CASE ID	**ITC2.1 :** This Should fail when using an invalid refresh token that doesn't exist in the table. **ITC2.2 :** This Should succeed when using a valid refresh token that exists in the table. **ITC2.3 :** This should fail when using a valid refresh token that is past expiry
Test case Summary	The Test case will fail if the refresh token does not exist in the database.
Prerequisites	The relevant functions are accessible and the path is proper. The database should be connected and running on Linux or WSL Portal. Also, all the relevant packages are installed. You can easily do this by the command `npm i`
Test Script/Procedure	Run the command `npm client.refreshtoken.test.js` from the test folder where all the test cases are implemented.
Test Data	An Invalid Refresh Token is provided.
Expected Result	The expected results are different based on different Refresh tokens. They include:

	`Invalid token. Please login with username and password again`
	`No refresh token found for client`
	`Token expired or revoked`

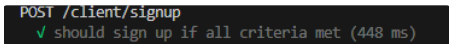
Actual Result	All the checks were done successfully in the test case. 
STATUS	**PASS**
Remarks	This checks with the Refresh tokens are working properly.
Created By	@Jai Phookan @Vijay Venkatesh
Executed by	@Jai Phookan @Vijay Venkatesh
Latest Date of Execution	May 2, 2024

3.) Client Validations - SignUp.

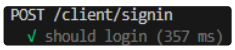
TEST SUITE ID	**ITCS3**
TEST CASE ID	**ITC3.1 :** Signup for the users on the portal.
Test case Summary	The Test case checks if the user can signup on the Portal.
Prerequisites	The relevant functions are accessible and the path is proper. The database should be connected and running on Linux or WSL Portal. Also, all the relevant packages are installed. You can easily do this by the command `npm i`
Test Script/Procedure	Run the command `npm client.test.js` from the test folder where all the test cases are implemented.

Test Data	<pre>1 username: "Different",
	2 first_name: "John",
	3 last_name: "Doe",
	4 email: "john.doe1@example.com",
	5 institution_name: "University A",
	6 pronoun: "he/him",
	7 password: "password@D5"</pre>
	This is the format of the data that was to be used. We can add multiple users.

Expected Result	`[200, 200, 500, 500]`
	<pre>1 [
	2 "",
	3 "",
	4 "Duplicate entry 'TestUser1' for key 'clients.username'",</pre>

	5 "Duplicate entry 'john.doe1@example.com' for key 'clients.emai 6];
Actual Result	All the checks were done successfully in the test case. 
STATUS	**PASS**
Remarks	The signup works for multiple conditions for the users.
Created By	@Vijay Venkatesh
Executed by	@Vijay Venkatesh
Latest Date of Execution	May 2, 2024

4.) Client Validations - Login.

TEST SUITE ID	**ITCS3**
TEST CASE ID	**ITC3.2 :** Login for the users by checking the password and the username.
Test case Summary	The Test case checks if the user can Login Correctly on the Portal.
Prerequisites	The relevant functions are accessible and the path is proper. The database should be connected and running on Linux or WSL Portal. Also, all the relevant packages are installed. You can easily do this by the command `npm i`
Test Script/Procedure	Run the command `npm client.test.js` from the test folder where all the test cases are implemented.
Test Data	1 username: 'sample_editor', 2 first_name: 'Number 1', 3 last_name: 'Dubakoor', 4 email: 'sample_editor@student.unimelb.edu.au', 5 institution_name: 'Uni', 6 orcid: 'orcid', 7 pronoun: 'other', 8 salt: '$2b$10$bJZV.MMykqNzLuqxw/EwGO', 9 password_hash: '$2b$10$bJZV.MMykqNzLuqxw/EwGOYutUiV9/r7VehUZsaVfa 10 email_verified: 1 This is the format of the data that was to be used.
Expected Result	• Should login successfully with valid credentials. • Should fail to login due to non-existent username. • Should fail to login due to wrong password
Actual Result	All the checks were done successfully in the test case. 
STATUS	**PASS**

Remarks	The Login works for multiple conditions for the users.
Created By	@Saurabh Zingade
Executed by	@Saurabh Zingade
Latest Date of Execution	May 2, 2024

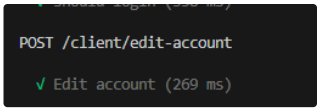
5.) Client Validations - Edit Account.

TEST SUITE ID	**ITCS3**
TEST CASE ID	**ITC3.3 :** Edit Account details for the user.
Test case Summary	The Test case checks if the user can signup on the Portal.
Prerequisites	The relevant functions are accessible and the path is proper. The database should be connected and running on Linux or WSL Portal. Also, all the relevant packages are installed. You can easily do this by the command `npm i`
Test Script/Procedure	Run the command `npm client.test.js` from the test folder where all the test cases are implemented.

Test Data	<pre>first_name: "sampleEditor",
	last_name: "new last name",
	institution_name: "Uni",
	pronoun: "he/him",
	-------------------------------- 	
	first_name: "",
	last_name: "new last name",
	institution_name: "Uni",
	pronoun: "he/him"
	-------------------------------- 	
	first_name: "First Name",
	last_name: "",
	institution_name: "Uni",
	pronoun: "he/him"
	-------------------------------- 	
	first_name: "First Name",
	last_name: "Last name",
	institution_name: "",
	pronoun: "he/him"</pre> This is the format of the data that was to be used.

Expected Result	<pre>"Updated account details",
	"First name is mandatory",
	"Last name is mandatory",
	"Institution name is mandatory"</pre> Depending on how the data is processed you get one of the results.

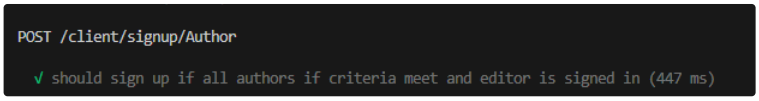
Actual Result	All the checks were done successfully in the test case.

	
STATUS	**PASS**
Remarks	The Edit works for multiple conditions for the users.
Created By	@Saurabh Zingade
Executed by	@Saurabh Zingade
Latest Date of Execution	May 2, 2024

6.) Client Validations - Author Signup By Editor.

TEST SUITE ID	**ITCS3**
TEST CASE ID	**ITC3.4 :** Should sign up if all authors if criteria meet and editor is signed in
Test case Summary	The Test case checks if a author can be assigned by an Editor.
Prerequisites	The relevant functions are accessible and the path is proper. The database should be connected and running on Linux or WSL Portal. Also, all the relevant packages are installed. You can easily do this by the command `npm i`
Test Script/Procedure	Run the command `npm client.test.js` from the test folder where all the test cases are implemented.

Test Data	`username: "TestUser5",`
	`first_name: "John",`
	`last_name: "Doe",`
	`email: "john.doe1@example.com",`
	`institution_name: "University A",`
	`pronoun: "he/him"`
	This is the format of the data that was to be used.

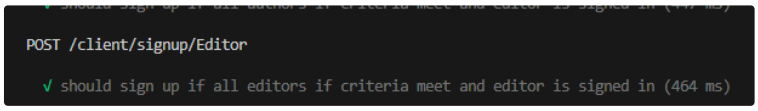
Expected Result	• Should Be assigned if all information is correct and non repetitive. • Should not work if there are duplicate entries.
Actual Result	All the checks were done successfully in the test case. 
STATUS	**PASS**
Remarks	The Author can be signed up by the editors.
Created By	@Vijay Venkatesh

Executed by	@Vijay Venkatesh
Latest Date of Execution	May 2, 2024

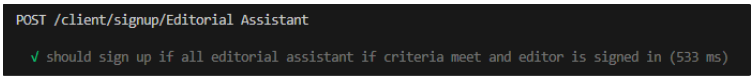
7.) Client Validations - Editor Signup.

TEST SUITE ID	**ITCS3**
TEST CASE ID	**ITC3.5 :** Should sign up if all editors if criteria meet and editor is signed in
Test case Summary	The Test case checks if a editor can access the platform information.
Prerequisites	The relevant functions are accessible and the path is proper. The database should be connected and running on Linux or WSL Portal. Also, all the relevant packages are installed. You can easily do this by the command `npm i`
Test Script/Procedure	Run the command `npm client.test.js` from the test folder where all the test cases are implemented.

Test Data	`username: "TestUser5",`
	`first_name: "John",`
	`last_name: "Doe",`
	`email: "john.doe1@example.com",`
	`institution_name: "University A",`
	`pronoun: "he/him"`
	This is the format of the data that was to be used.

Expected Result	• The Editor should be assigned if the data is correct.
Actual Result	All the checks were done successfully in the test case. 
STATUS	**PASS**
Remarks	The Editors can be assigned correctly.
Created By	@Vijay Venkatesh
Executed by	@Vijay Venkatesh
Latest Date of Execution	May 2, 2024

8.) Client Validations - Editorial Assistant Signup.

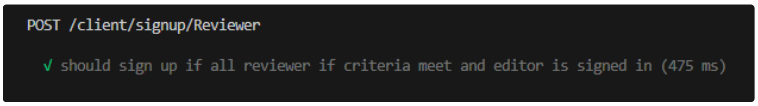
TEST SUITE ID	ITCS3
TEST CASE ID	**ITC3.6 :** Should sign up if all editorial assistant if criteria meet and editor is signed in
Test case Summary	The Test case checks if an editorial assistant can access the platform information.
Prerequisites	The relevant functions are accessible and the path is proper. The database should be connected and running on Linux or WSL Portal. Also, all the relevant packages are installed. You can easily do this by the command `npm i`
Test Script/Procedure	Run the command `npm client.test.js` from the test folder where all the test cases are implemented.
Test Data	`username: "TestUser5", first_name: "John", last_name: "Doe", email: "john.doe1@example.com", institution_name: "University A", pronoun: "he/him"` This is the format of the data that was to be used.
Expected Result	• The editorial should be assigned if the data is correct.
Actual Result	All the checks were done successfully in the test case. 
STATUS	PASS
Remarks	The Editorial Assistants can be assigned correctly.
Created By	@Vijay Venkatesh
Executed by	@Vijay Venkatesh
Latest Date of Execution	May 2, 2024

9.) Client Validations - Reviewer Signup.

TEST SUITE ID	ITCS3
TEST CASE ID	**ITC3.7 :** Should sign up if all reviewer if criteria meet and editor is signed in
Test case Summary	The Test case checks if an editorial assistant can access the platform information.

Prerequisites	The relevant functions are accessible and the path is proper. The database should be connected and running on Linux or WSL Portal. Also, all the relevant packages are installed. You can easily do this by the command `npm i`
Test Script/Procedure	Run the command `npm client.test.js` from the test folder where all the test cases are implemented.

Test Data	<pre>1 username: "TestUser5",
	2 first_name: "John",
	3 last_name: "Doe",
	4 email: "john.doe1@example.com",
	5 institution_name: "University A",
	6 pronoun: "he/him"</pre> This is the format of the data that was to be used.

Expected Result	• The reviewer should be assigned if the data is correct.
Actual Result	All the checks were done successfully in the test case. 
STATUS	**PASS**
Remarks	The Reviewers can be assigned correctly.
Created By	@Vijay Venkatesh
Executed by	@Vijay Venkatesh
Latest Date of Execution	May 2, 2024

Some comments have been generated using ChatGPT.

SDLC Model

The Agile Software Development Lifecycle (SDLC) has been adopted for our project. Agile methodology provides a flexible and iterative approach to software development, emphasizing collaboration, adaptability, and delivering incremental value to stakeholders.

Release Cycles

Our project follows a series of release cycles, each comprising 1 sprint aimed at delivering valuable features. These release cycles are structured to align with project milestones and customer expectations. Each release cycle culminates in the delivery of a potentially usable product increment, providing stakeholders with tangible progress and opportunities for feedback.

Sprint Lengths

Sprint lengths have been chosen to be 4 weeks allowing for completion and quality testing of completed user stories. The project is estimated to be completed after 4 sprints overall. The first sprint being the Inception/Design sprint and the subsequent being the development sprints. A usable software entity will be provided to the client for feedback at the end of subsequent sprints.

Ceremonies

Throughout each sprint, the team engages in various ceremonies to facilitate collaboration, transparency, and continuous improvement. These ceremonies include:

Sprint Planning

- At the beginning of each sprint, the team conducts a sprint planning session to review the backlog, select items for implementation, and define the sprint goal and tasks. This ceremony ensures alignment among team members and clarity on sprint objectives.

Daily Stand-up Meetings

- Daily stand-up meetings are held to provide a forum for team members to synchronize their activities, discuss progress, identify impediments, and plan their work for the day. These short, focused meetings promote transparency, accountability, and collaboration within the team.

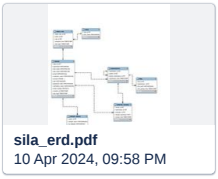
Sprint Review

- At the end of each sprint, a sprint review session is conducted to demonstrate the completed work to stakeholders, gather feedback, and assess the sprint's overall success. This ceremony enables stakeholders to provide input, validate deliverables, and ensure alignment with project goals.

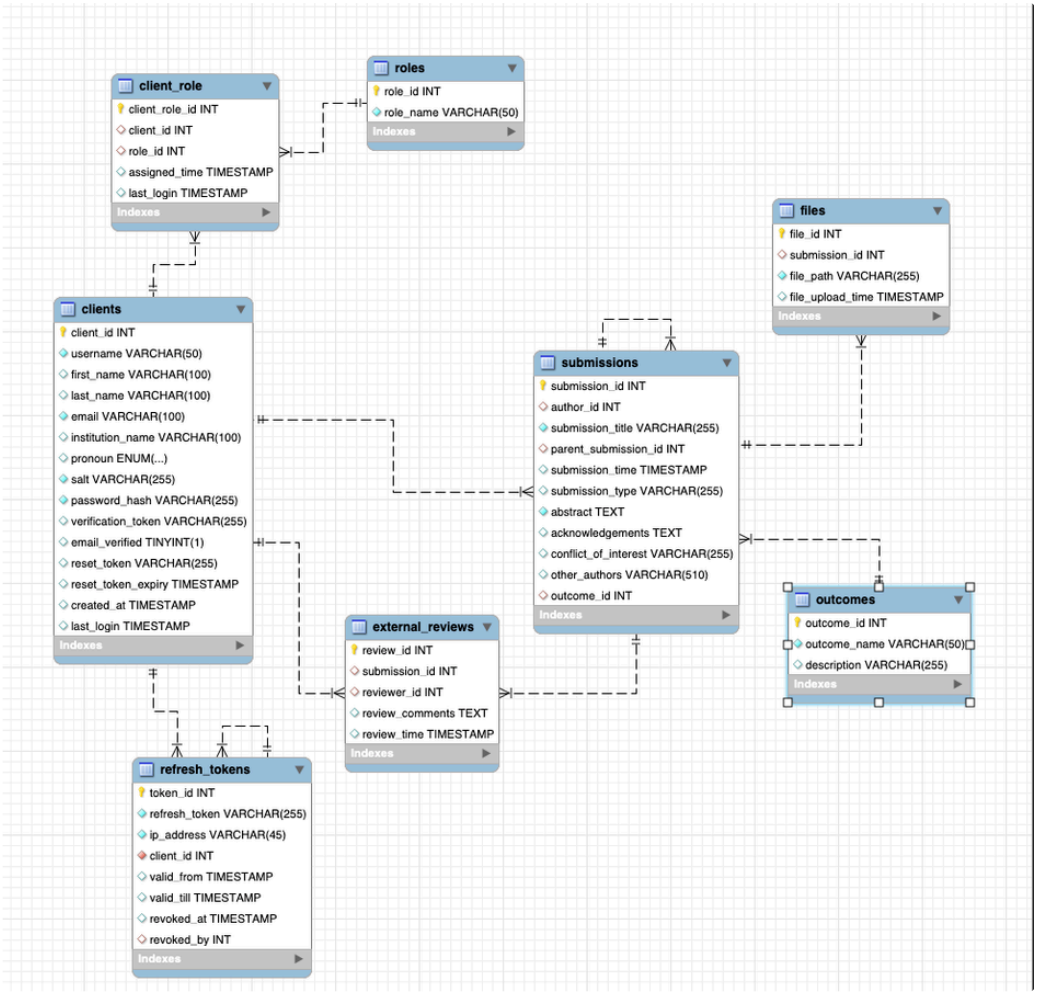
Sprint Retrospective

- Following the sprint review, the team conducts a sprint retrospective to reflect on the sprint's processes, identify areas for improvement, and establish action items for the next sprint. This retrospective ceremony fosters a culture of continuous learning and adaptation, empowering the team to enhance its performance and productivity over time.

SiLA ERD



sila_erd.pdf
10 Apr 2024, 09:58 PM



! Ethics Centred Development

Customer’s Right To Be Forgotten (a.k.a Right of Erasure)

Originally, stated in the European GDPR laws, which gives the the right to a customer to ask third party that is holding their personal data to remove it completely. Although, not explicitly stated in the Australian Privacy Principles, citizens don’t have the right to request parties to delete there data. However, several companies in Australia have been in the forefront of giving this privilege to its uses.

Given SiLA operates under complicated IP laws, it was only apt for us to design a system which safeguards the intellectual property of authors. As of today, if a client deletes their account in the system, all submissions and IP that they have submitted to SiLA will be redacted as well and at no point will SiLA hold any IP which does not belong to any recorded client (i.e. each piece of work should be able to be traced back to an active user).

Data Privacy

The Portal ensures that users are fully informed about how their data is collected, used, and stored. Obtain explicit consent for processing personal information. Only the data that is absolutely necessary is collected to avoid excessive data retention. Also, Data Security is a top priority. The way how data is stored is explained in detail over HERE.

Equally important is obtaining informed consent from participants in research endeavors involving their data. Informed consent entails transparently communicating the purpose, risks, and benefits of the study to participants, empowering them to make well-informed decisions about their involvement. This practice fosters trust and respect for individuals' autonomy while upholding ethical standards in research. Ultimately, ethical use of data analytics and research hinges on balancing the pursuit of valuable insights with a steadfast commitment to protecting user privacy and ensuring informed consent at every stage of the process.

Diversity and Inclusion

In the realm of Diversity and Inclusion, it is imperative to uphold principles of non-discrimination and inclusive design. Non-discrimination entails ensuring that every facet of data collection, processing, and decision-making remains impartial, devoid of biases related to race, ethnicity, gender, sexual orientation, religion, or any other protected characteristics. This commitment translates into fair treatment for all individuals, fostering an environment where everyone feels valued and respected.

Simultaneously, inclusive design becomes paramount in crafting products and services that cater to diverse user needs. By considering accessibility requirements and cultural sensitivities during the design phase, SiLA can create solutions that are accessible and welcoming to a wide spectrum of users. This approach not only enhances user experience but also reflects a commitment to inclusivity, recognizing the richness of diversity within our communities and acknowledging the importance of equitable access to technology and services.

SOME SENTENCES ARE REWRITTEN USING CHATGPT.

🗝️ Security

CORS

In the backend system, CORS settings dictate which origins are authorized to access server resources. Currently, our configuration permits access from local development servers and the frontend deployment origin exclusively. This strict control ensures that resources are shared only with trusted sources, safeguarding against unauthorized cross-origin requests and bolstering the overall security of our web applications.

Intention to use https during deployment

HTTPS encrypts data transmitted between the client (such as a web browser) and the server, making it unreadable to anyone who might intercept it. This protects sensitive information such as passwords and tokens shared. This also ensures data integrity that the transmitted data is unaltered and mitigates the risk of man-in-the-middle attacks. We intend to integrate SSL certificates before the final deployment.

Password storage

Regarding password storage, we employ a robust method using salt and hash techniques to securely store user passwords in our database. Utilizing the 'eksblowfish' methodology, passwords are transformed into irreversible hashes, ensuring that actual passwords are never exposed during transmission. This one-way hashing function guarantees that original passwords cannot be reverse-engineered from the stored hashes.

Token system for role and permission management

For role and permission management, we utilize a token-based system employing JSON Web Tokens (JWTs). The predefined roles being:

- Editor
- Editorial Assistant
- Author
- Reviewer

Access to specific APIs is granted based on the roles associated with the user. This information is embedded within the json web token. This approach facilitates the implementation of a comprehensive role-permission system, ensuring that unauthorized access and operations are prevented effectively.

Environment Configuration Management

- All sensitive information like username, password, etc are passed to the builds via `.env` files or embedded terminal variables.

Sprint Planning

The Following Section contains an Overview of how the Sprints will be delivered and what can be expected at the end of each sprint. Each section has section about what is planned and the expected delivery date.

Sprint 1: Start(Design Sprint)

Monday 5th March to Friday 22nd March

COMPLETED TASKS:

1.) Understanding Key Requirements: Analysed the Document given by the Clients to understand what exactly is needed at the end of the project.

2.) Role Assignment and Project OwnerShip: Roles were assigned to different team members based on their skills. It was decided that the team would be cross-functional and would be adjusted based on the requirements.

3.) Setting up Epics and User Stories: The team set up the Jira page along with the timelines for different epics and the user stories associated with those epics.

4.) Tech-Stack Discussion: Used React.js for frontend, Node.js for backend, and MYSQL as the database.

5.) Documentation and Rules:

- Set up the Documentation and the Confluence page for the project.
- Created and established Github Development rules for proper code management.
- Established a good communication channel on Slack for the entire team.

6.) Meeting Establishments: Decided on a time to meet in person every week at least once which is Wednesday at 11 pm at The University of Melbourne campus.

Sprint 2: Delivery Phase 1

26th March Monday to 26th April Friday

PLANNED TASKS:

1.) FrontEnd: We plan to finish the User registration epic which contains all the user registration tasks. We also plan to develop the feature for the Manuscript submission which is another epic in this Sprint on the FrontEnd aspect.

2.)BackEnd: The database setup will be done and all the data from the front end will be processed and all the relevant data will be shown to the user from the backend.

3.) Sprint Retrospective and Future Sprint planning: In the end, we plan to check how this Sprint progressed and what exactly we will be doing in the next Sprint.

Sprint 2: Delivery Phase 2

30th April Monday to 26th May Friday

PLANNED TASKS:

1.) FrontEnd: This Sprint will mostly focus on working on the editorial decision-making tasks.

2.)BackEnd: The backend will be used to assign proper roles to the users based on their profiles stored in the database. The Data Security and Confidentiality Epic will be developed during this sprint.

3.) Sprint Retrospective: In the end, we plan to check how this Sprint progressed and what further final requirement changes are needed by the Client.

Sprint 4: Product Handover

1st May Monday to 5th June Friday

PLANNED TASKS:

Handover the Documentation and the final product to the client along with the changes mentioned by the client after the end of Sprint 4.

Project Plan - Timeline

2 Story Points = 1 Day's Work Per Person

COMP90082-2024-SI-KOALA

Client must be authenticated to access this resource.

1 Sprint 1 Deliverables

START DATE: Monday 5th March

END DATE: Friday 22nd March

Completed Work:

Task ID	Task	Status
T1	Understanding the User Requirements	COMPLETE
T2	Distribution of the workflow among the Team	COMPLETE
T3	Setting up the Confluence Documentation	COMPLETE
T4	Set up for the GitHub Repository	COMPLETE
T5	Create the Personas on the Confluence	COMPLETE
T6	Jira board setup with epics and User stories	COMPLETE
T7	Note down all the meeting minutes on Confluence	COMPLETE
T8	Note down the Goal model on Confluence	COMPLETE
T9	Generate the Submission tag on Github for Submission	COMPELTE

Note: These tasks are not mentioned on the Jira board for the project.

Project Progress Update 28 March, 2024

Overview of Work

User Registration & Management

- Secure Login 👷
- Profile Tracking 👷

Manuscript Submissions

- UI for authors 👷
- Email confirmation 🗓️

Manuscript Tracking & Management

- Dashboard to track submissions 🗓️
- Delegation of review 🗓️
-

Peer Review System

- Double-blind review 🗓️
- Communication module ❌
- Feedback via template 👷

Editorial Decision-Making Tools

- Record decisions 🗓️
- Notification service 🗓️

Data Security & Confidentiality

- Confidentiality of manuscripts and reviews 👷
-

Scalability & Flexibility

- Volume ❌
- Variety 👷

Integration Capabilities

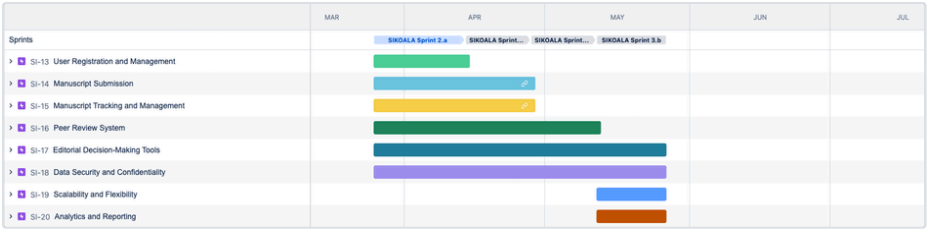
- Academic indexing integration ❌
-

Analytics & Reporting

- Get insights on submissions 🗓️

ℹ️
✅ - Completed
❌ - De-prioritised
👷 - Work In Progress
🗓️ - Planned

Timelines



Sprint Name	From	To
Sprint 2.a	Mar 25, 2024	Apr 5, 2024
Sprint 2.b	Apr 8, 2024	Apr 26, 2024
Sprint 3.a	Apr 29, 2024	May 10, 2024
Sprint 3.b	May 13, 2024	May 31, 2024

🙌 Sprint 1 Review

What🗓️ Date

Mar 22, 2024

👥 Participants

- @Saurabh Zingade
- @Vijay Venkatesh
- @Rishabh Srivastava
- @Jai Phookan
- @Akash Renuka Ashok
- @xiyanedwinz

🥅 Goals

- Identify what Went Right
- Review the Sprint Goal.
- Address the Weakness and Potential Improvements
- Communication discussion
- Code Review Discussion
- Set Actionable Goals
- Encourage Learning
- Build the Team Strength.

🗣️ Discussion Topics

1.) **Ownership Distribution:** Discussed and assigned ownership for different project aspects.

- FrontEnd : @Akash Renuka Ashok @Saurabh Zingade
- BackEnd : @Vijay Venkatesh @Jai Phookan @Saurabh Zingade
- DataBase: @xiyanedwinz @Rishabh Srivastava

2.) **Managing Dependencies**: Identified and prioritized dependencies between epics.

3.) **Tech Stack Decision**: Finalized tech stack: React.js for Frontend, Node.js for Backend, and MySQL for Database.

4.) **Sprint Planning**: Assigned epics to specific sprints and finalized deliverables.

5.) **Sprint Execution:** Confirmed commitment to achieving Sprint deliverables as planned.

6.) **Deployment Strategy:** Agreed upon local deployment and MySQL dump file management for database spin-up. Later we will deploy the application on AWS servers.

✅ Action items

- [x] Create User tasks as sub-tasks for User stories.
- [x] Set up initial Frontend setup and push to GitHub.
- [x] Schedule another meeting before Sprint 2 to review tasks for User stories.
- [x] Update the Jira board regularly.
- [x] Answer the Polly on Slack mentioning about what you were working on during the previous day.
- [x] Github contribution and merge policy.
- [x] Sprint 2 planning znd ownership allocation.

Decisions

- Stick to planned story points for each Story.
- Ownership model finalized.
- Tech stack decision: React.js (Frontend), Node.js (Backend), MySQL (Database).
- Sprint planning for Sprint 2 will be completed with assigned deliverables and timelines.
- Encourage ongoing feedback and continuous improvement.
- Communicate action items and decisions to the entire team.

Some of the sentences are grammatically corrected by ChatGPT on this page.

2 Sprint 2

2 Story Points = 1 Day's Work Per Person

The following User Stories are currently planned for Sprint 2. We add tasks when we start developing each of the User Stories. All the Sub-Tasks are available here.

Type	Key	Summary	Story point...	Sprint	Assignee
	SI-32	As an editor, I want to communicate the outcome of ...	14	SIKOALA Sprint 2.b SIKOALA Sprint 2.a	Akash Renuka
	SI-31	As an editor, I want to see progress of submissions ...	10	SIKOALA Sprint 2.b SIKOALA Sprint 2.a	Akash Renuka
	SI-30	As an author, I want a simple and easy to use UI so...	8	SIKOALA Sprint 2.b SIKOALA Sprint 2.a	Jai Phookan
	SI-28	As an author, I want to be able to submit a manuscr...	10	SIKOALA Sprint 2.b SIKOALA Sprint 2.a	Jai Phookan
	SI-23	As an editor/editor-assistant, I want to create accou...	2	SIKOALA Sprint 2.b SIKOALA Sprint 2.a	Saurabh Zinga
	SI-22	As an author, I want to be able to signup on the plat...	6	SIKOALA Sprint 2.a	Saurabh Zinga
	SI-21	As a user, I want to be able to securely login to the ...	10	SIKOALA Sprint 2.b SIKOALA Sprint 2.a	Saurabh Zinga
	SI-33	As an editor, I should be able to desk reject so that ...	8	SIKOALA Sprint 2.b	Saurabh Zinga
	SI-29	As an author, I want to be able to receive an email ...	8	SIKOALA Sprint 2.b	Jai Phookan
	SI-25	As a user, I want to be able to edit my account detai...	8	SIKOALA Sprint 2.b	Saurabh Zinga
	SI-24	As a user, I want to be able to reset my credentials ...	5	SIKOALA Sprint 2.b	Akash Renuka

11 items

Synced just now

Subtasks for Sprint 2

We divided Sprint 2 in 2 different parts.

1.) Sprint 2.a for the first 3 weeks.

2.) Sprint 2.b for the next 2 weeks.

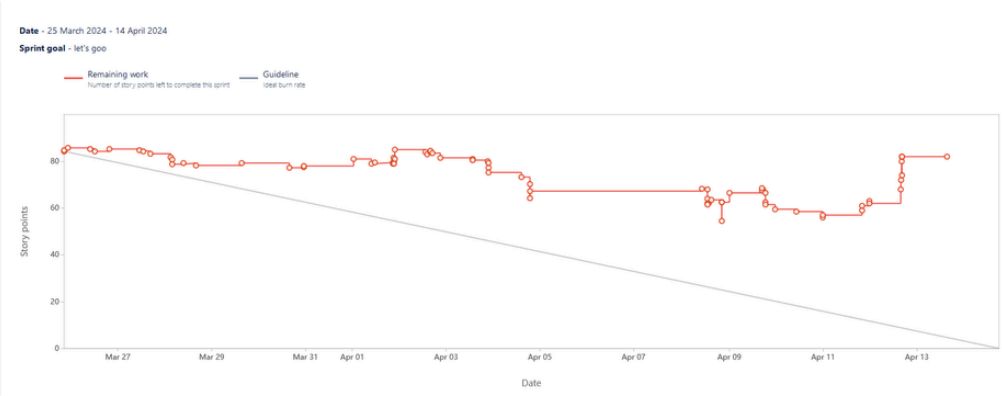
This was done for better management. We have already conducted the Retrospective for the Sprint which can be reviewed here. The Retrospective for Sprint 2.b and The overall Sprint 2 as well as the Review for the Sprint will be conducted soon as the Sprint Concludes.

Sprint 2.a Completed tasks are in the table below. The Tasks can be accessed through the team's Jira Board here.

Task Number	Title	**Story Points**	Assignee
SI-44	**Create Client Table**	0.5	@Rishabh Srivastava
SI-45	Create a Role Table	0.5	@xiyanedwinz
SI-46	Create user-role association table	0.5	@Rishabh Srivastava
SI-47	Create Submission table	1	@xiyanedwinz
SI-48	Create a review table	1	@Rishabh Srivastava
SI-50	Submit file API	1	@Rishabh Srivastava
SI-52	Create an account	NA	Not needed anymore
SI-53	Create Outcome table	0.5	@xiyanedwinz
SI-61	Signup and Login Page	2	@Saurabh Zingade
SI-62	Authentication API	1	@Vijay Venkatesh
SI-63	Forgot Password API	2	@Jai Phookan
SI-65	Password Policy	1	@Vijay Venkatesh
SI-70	Manuscript Form for Manuscripts	1	@Akash Renuka Ashok
SI-71	UI for the Reviewer to perform the tasks.	1.5	@Akash Renuka Ashok
SI-72	Signup API	2	@Vijay Venkatesh
SI-77	Progress bar for Author	2	@Akash Renuka Ashok
SI-77	Progress bar for Editor	0.5	@Akash Renuka Ashok
SI-84	Auto email Sending Functionality	2	@Jai Phookan
SI-85	Gmail Service account	0.25	@Jai Phookan
SI-89	Setuoof Routing Components.	1	@Saurabh Zingade
SI-97	Windows Database Setup	1	@Vijay Venkatesh
SI-92	Validate API for Signup	0.75	@Vijay Venkatesh
SI-93	Link Signup API	3	@Saurabh Zingade

SI-95	Token System Implementation	4	@Vijay Venkatesh
SI-96	Verify Email During SIgnup	2	@Vijay Venkatesh
SI-97	Implement cascading for table	1	@Rishabh Srivastava
SI-98	Create JWT tables	0.5	@Rishabh Srivastava
SI-101	Forgot password API Link	2	@Saurabh Zingade
SI-102	Different Login and Signup pages	1	@Saurabh Zingade
SI-103	Edit Details API	2	@Vijay Venkatesh

The following diagram is the Sprint burndown chart for the Sprint 2.a.



Burndown for Sprint 2.a

More Information HERE.

We tried to improve this in the following Sprint 2.b based on the Retrospective meeting for the initial Sprint.

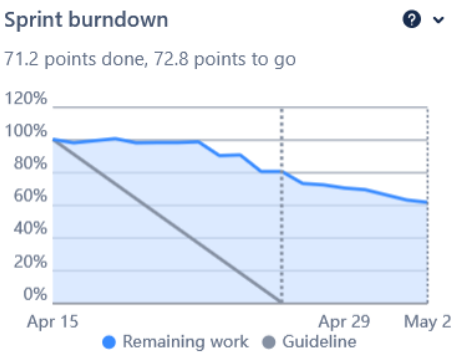
These are the tasks that are completed in Sprint 2.b until April 30th.
SOme of the tasks might be added later on which might not be reflected in the table below. The latest status can be accessed through the JIRA.

Task Number	Title	**Story Points**	Assignee
SI-67	UI for Author	1	@Saurabh Zingade
SI-69	UI for Editor to assign Reviewer.	1	@Saurabh Zingade
SI-80	UI for Editorial Assistant to assign Editors.	1	@Saurabh Zingade
SI-86	ManuScript Submission	2	@Jai Phookan
SI-100	Login API link to Frontend	2	@Saurabh Zingade
SI-104	Deplot Infrastructure	4	@Rishabh Srivastava
SI-105	GitHub Actions	2	@Rishabh Srivastava
SI-106	Protected Routes	1	@xiyanedwinz
SI-109	Edit Information UI	1	@Saurabh Zingade

SI-116	Test Case For password policy	0.5	@Vijay Venkatesh
SI-117	Unit test case for Email	0.5	@Saurabh Zingade
SI-118	Unit Test case for managing Token	0.5	@Rishabh Srivastava
SI-130	Find Orignal Submission	0.5	@Rishabh Srivastava
SI-131	UI for Reviewer Form	2	@Saurabh Zingade
SI-137	Fix MYSQL vulnerability	0.25	@Vijay Venkatesh
SI-138	JEST setup	2	@Vijay Venkatesh
SI-140	S3 Upload for Frontend	0.5	@Rishabh Srivastava
SI-141	Find latest Submission	2	@xiyanedwinz
SI-142	Setup Test DB	0.5	@Rishabh Srivastava
SI-143	Review Form in Database	1	@xiyanedwinz
SI-144	Setup Test Workflow	1	@Rishabh Srivastava
SI-145	Cleanup test db stored Procedure	1	@xiyanedwinz
SI-147	Authorisation using token	0.5	@Vijay Venkatesh
SI-148	Autoemal confirmation after submission	1	@Jai Phookan
SI-154	Database Requirements	1	@Rishabh Srivastava
SI-155	Evaluate FK constraints	0.5	@xiyanedwinz
SI-156	Test case for Email validation	0.5	@Saurabh Zingade
SI-157	Generate password Testcase	1	@xiyanedwinz
SI-158	Integration testing for Edit user	1	@Saurabh Zingade
SI-159	Integration test for Signup	1	@Vijay Venkatesh
SI-161	Integration test for signup for Editor	1	@Vijay Venkatesh
SI-162	Signup and Signin Fixes	1	@Vijay Venkatesh
SI-166	Integraion Test for Mansucript Submission	1	@Jai Phookan
SI-167	Integraion Test for Refresh Token	1	@Jai Phookan
SI-171	Test case for Incomplete field	0.5	@Saurabh Zingade

The Burndown chart for Sprint 2.b is not done yet since the sprint is not yet complete.

At the time of the documentation this is how the Burndown chart looks for Sprint 2.b.


Sprint burndown
71.2 points done, 72.8 points to go
120%
100%
80%
60%
40%
20%
0%
Apr 15
Apr 29
May 2
Remaining work
Guideline

3 Sprint 3 Planning

2 Story Points = 1 Day's Work Per Person

The following User Stories are currently planned for Sprint 3. We will be adding new sub-tasks for each of the User stories.

Type	Key	Summary	Assignee	Status	Sprint
	SI-39	As an editor, I want authors to only see their manus...	Rishabh Srivastava	TO DO	SIKOALA Sprint
	SI-38	As an editor, I want to see the status of all submitte...	Vijay Venkatesh	TO DO	SIKOALA Sprint
	SI-37	As an editor, I want to be able to assign manuscript...	Vijay Venkatesh	TO DO	SIKOALA Sprint
	SI-35	As an editor, I want reviewers to share reviews in a ...	Akash Renuka Ashok	TO DO	SIKOALA Sprint
	SI-34	As a reviewer, I want to share my reviews to the edi...	Rishabh Srivastava	TO DO	SIKOALA Sprint
	SI-26	As a user, I want to be able to switch between assig...	Akash Renuka Ashok	TO DO	SIKOALA Sprint
	SI-42	As an editor, I should be able to view reports on cur...	xiyanedwinz	TO DO	SIKOALA Sprint
	SI-41	As a editor/editor-assistant, I want the platform to ac...	xiyanedwinz	TO DO	SIKOALA Sprint
	SI-40	As an editor, I want to make sure reviewers don't kn...	Rishabh Srivastava	TO DO	SIKOALA Sprint
	SI-36	As an editor, I want to be able to submit my decisio...	Vijay Venkatesh	TO DO	SIKOALA Sprint
	SI-27	As an editor/editor-assistant, I should be able to ass...	Akash Renuka Ashok	TO DO	SIKOALA Sprint

11 items

Synced just now

😀 Retro

- Retrospective: SIKOALA Sprint 2.a

🛠️ Retrospective: SIKOALA Sprint 2.a

📋 Overview

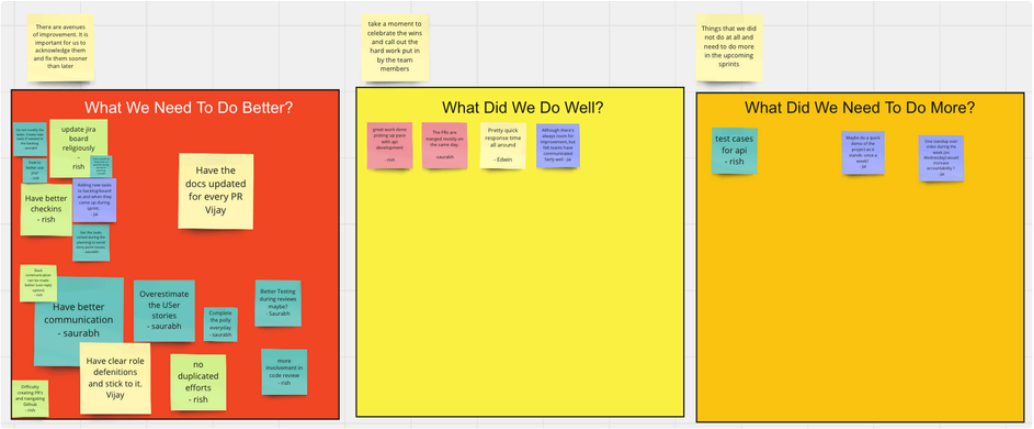
Reflect on past work and identify opportunities for improvement by following the instructions for the.

Date	Apr 12, 2024
Team	SI-Koala
Participants	@Akash Renuka Ashok @xiyanedwinz @Jai Phookan @Vijay Venkatesh @Saurabh Zingade @Rishabh Srivastava

💭 Retrospective

Start doing	Improvements	Keep doing

• Test cases
• Project showcases
• Weekly catchup
	• Communication
	  ◦ Slack
	  ◦ Github PR's
	  ◦ etc
	• Have clear roles and definitions
	• Keep documentation up to date.
	• Overestimate story points
	• Complete daily standup poll daily
	• Avoid duplicated efforts
	• Don't modify existing tasks, create new ones if needed
	• Update progress on Jira well
	• Add tasks before beginning work
		• Great work picking up pace
		• Quick turnaround for requested work



Miro Board https://miro.com/app/board/uXjVKWJlaf4=/ Connect your Miro account

Initial Thoughts On Similar Peer Review Software.

Contributors

@Akash Renuka Ashok

@Rishabh Srivastava

@Jai Phookan

@Vijay Venkatesh

@xiyanedwinz

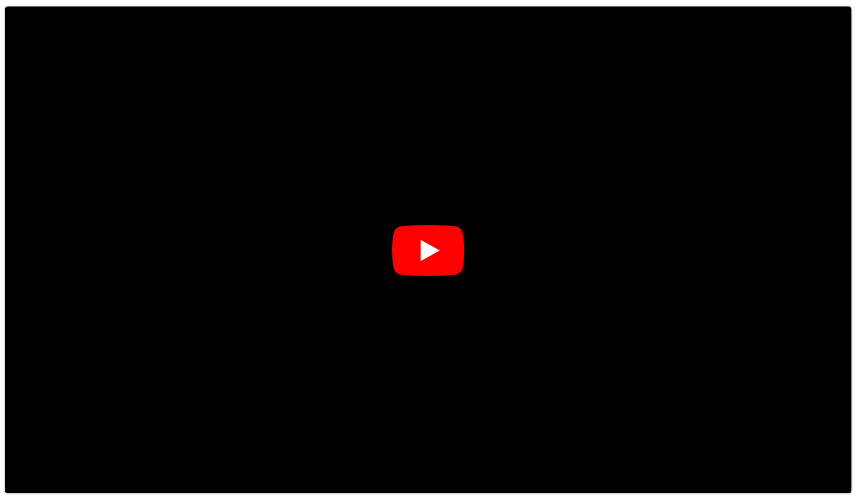
@Saurabh Zingade

The following are some of the similar platforms that we are trying to develop. So we might use them for capturing some ideas.

1. Microsoft CMT

The Video gives a good overall overview of the platform :

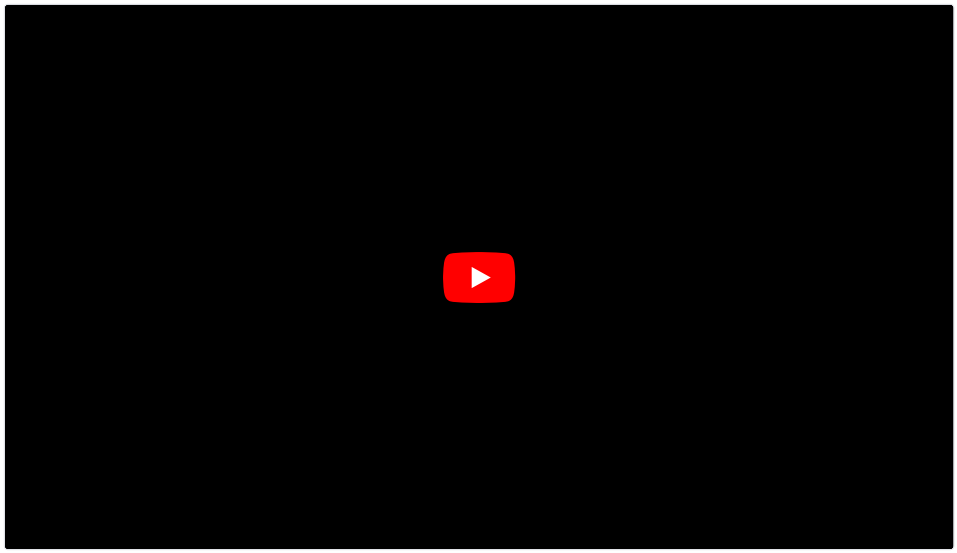
Microsoft CMT Registration and Submission Creation at FES Congress



2. Easy Chair

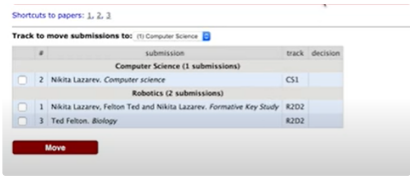
Followed this tutorial series to get a sense of it

1. Logging into EasyChair and Finding the Submission Link - Tutorial

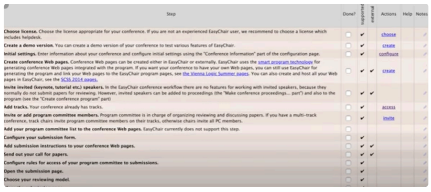


Key features to consider:

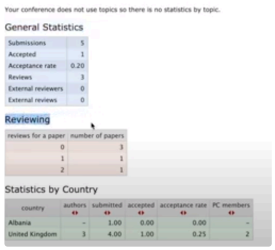
- Submission tab - straightforward process, inputting information and upload file button
  - Submissions then added to a page that showcases all submissions.
  - Submissions can be moved to a different “track” (error correcting)



- Admin role - like an option page to setup new conferences (new papers/editions/special editions)



- Accepting and rejecting papers
  - Overall editior’s role to reject or accept
  - can accept and reject from dashboard
- Assigning reviews/requesting reviews
  - Editor requests specific reviewers
- Adding reviews
  - Review’s “vote” on accepting and rejecting (-3 to strongly reject, +3 for strongly accept)
  - Reviewers add confidence level, editor can take level of expertise into account
- Authors recieve notification when reject/accept decisons recieved
  - Authors can see the reivews they recieved
- Editors can invite new users to the webpage, users create own new accounts, request clearance levels, editors can change role of users
- Setting conflicts of interest
  - Adding conflicts of interest (editor or users can too?)
- Conference settings



🗒️ Template - Meeting notes

🗓️ Date

👥 Participants

-
-

🥅 Goals

-

🗣️ Discussion topics

Time	Item	Presenter	Notes
			•

✅ Action items

- [ ]

⤴️ Decisions

↙️

✍️ Template - Product requirements

Target release	Type // to add a target release date
Epic	Type /Jira to add Jira epics and issues
Document status	DRAFT
Document owner	@ mention owner
Designer	@ designer
Tech lead	@ lead
Technical writers	@ writers
QA	

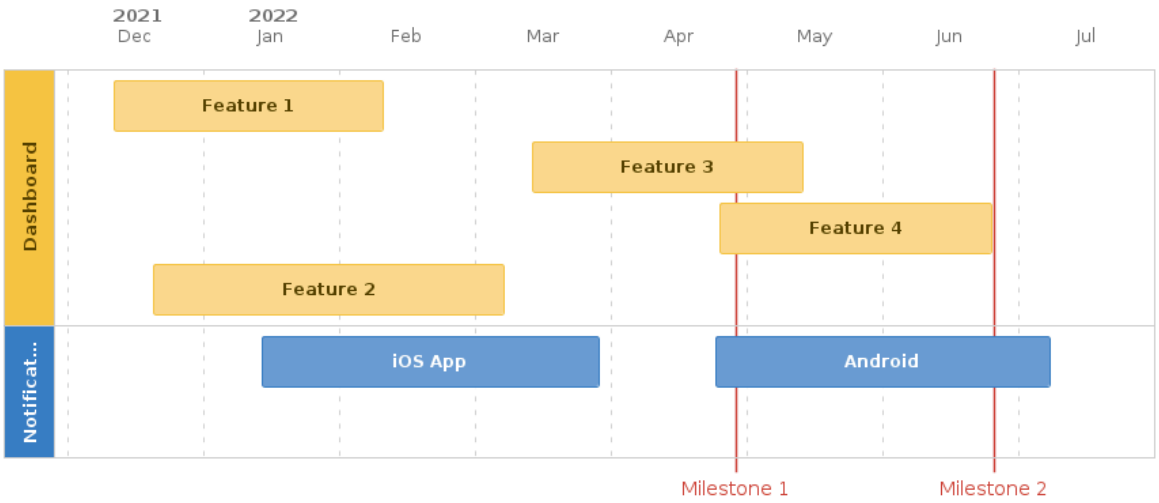
🎯 Objective

📊 Success metrics

Goal	Metric

🤔 Assumptions

🌟 Milestones



🗒️ Requirements

Requirement	User Story	Importance	Jira Issue	Notes
		HIGH		

🎨 User interaction and design

❓ Open Questions

Question	Answer	Date Answered

⚠️ Out of Scope

-

✅ Template - Decision documentation

Status	NOT STARTED / IN PROGRESS / COMPLETE
Impact	HIGH / MEDIUM / LOW
Driver	
Approver	
Contributors	
Informed	
Due date	
Resources	

📚 Relevant data

📘 Background

🌈 Options considered

	Option 1	Option 2
Description		
Pros and cons	➕ ➖	➕ ➖
Estimated cost	LARGE	MEDIUM

✅ Action items

- [ ]

🌟 Outcome

Retrospectives

Add Retrospective

Title	Date	Participants
Retrospective: SIKOALA Sprint 2.a	Apr 12, 2024	@Akash Renuka Ashok @xiyanedwinz @Jai Phookan @Vijay Venkatesh @Saurabh Zingade @Rishabh Srivastava

Meeting notes in space

Create meeting note

Incomplete tasks from meetings

Task report

Looking good, no incomplete tasks.

Decisions from meetings

Page Title	Decisions
Team Meeting No 1: 8th March	The team got to know more about each other.
Team Meeting No 2 : 13th March	When we start working on the Epics there might we some more user stories that will be added based on the needs. The team analyzed the four main personas while evaluating each User Story. 1.) Author 2.) Editor 3.) Reviewer 4.) Editorial Assistant. Every Team member is responsible for assigning one epic and needs to add story points to all the Items within that particular Epic. We need to create sub-tasks when we assign the user stories to someone. The person who the user story is assigned to is also responsible for creating the proper sub-tasks for that particular User Story. The workflow is distributed as Follows for the following week as shown in the image below

All meeting notes

Title	Creator	Modified
Team Meeting No 9: 19th April	Vijay Venkatesh	Apr 19, 2024
Team Meeting No 2 : 13th March	Saurabh Zingade	Mar 16, 2024
Team Meeting No 1: 8th March	Rishabh Srivastava	Mar 16, 2024