1.1 Software Requirements Specification (SRS)

Problem statement

The current ethics approval process for research projects is a online form which is inefficient. Researchers submit the forms which doesn't fill the needs requirement documents, which causes delays, missing attachments, and lack of transparency.

Objectives

1. Digitalize the ethics application process.
2. Enable researchers to submit and track applications online.
3. Automate notifications and reminders for pending approvals.
4. Provide an approval acurate workflow.
5. Maintain a secure archive of all submissions.

Functional Requirement

Researcher can:

- Create and submit ethics application form online.
- Upload required documents (proposal, consent forms, etc.).
- Receive system notifications (approval, rejection, modification requests).
- Communicate with reviewers via the built-in messaging system.

Faculty Reviewer can:

- Review and forward applications to the Ethics Committee.
- Request modifications.

Ethics Committee can:

- Approve, reject, or request revisions.
- Add comments or notes for applicants.

Admin can:

- Manage users, roles, and approvals.
- Receive reminder after 48 hours for unattended applications.
- Configure system notification interval (Super Admin).
- KPIs view to track all the operation of the platform

- Possibility of upgrading or downgrading the role of a user
- Possibility to track the process history of an application (searching area)
- possibility of assigning the application for other approvals (during the process)
- Area where we can define workflows: instead of code implementation and for more flexibility, the admins should be able to customise different workflows for different applications and assign them to the form applications.

System automatically:

- Prevents incomplete submissions. (provide confirmation messages for onEdit applications: implement a confirmation message dialog instead of using the browser message)
- Tracks all actions (audit logs).
- Stores documents securely in cloud storage: define a storage method for data privacy)

Non functional Requirement

- Performance: Handle more users.
- Availability: 99.9% uptime.
- Security: Authentication , encrypted file uploads.
- Usability: Simple web-based interface (React).
- Scalability: Modular backend (Node.js + PostgreSQL).
- Maintainability: RESTful APIs, well-documented code.

System user and role

Role	Description	Permissions
Researcher	Submit applications and upload files	Create, Read, Edit
Faculty Reviewer	Review and forward	Read, comment & request for updates
Ethics Committee	Approve/Reject/Modify	Full review
Admin	System oversight	Crud on records
Super Admin	Configuration	Full access