ReMa - Research Material Submission Tracker

Repository Structure

.idea

- .gitignore
- modules.xml
- Submission_Tracker.iml
- vcs.xml

public

- prof_script.js

SQL

- Data Population.sql
- Database_schemas.sql

src

- routes
    - prof_routes
        - conferences.js
        - deadlines.js
        - mail.js
        - prof_dashboard.js
        - reading_list.js
        - research_projects.js
        - sponsored_projects.js
        - students.js
        - todo.js
        - under_review_papers.js
    - student_routes
        - conferences.js
        - deadlines.js
        - profs.js
        - reading_list.js
        - research_projects.js
        - student_dashboard.js
        - todo.js
        - under_review_papers.js
    - admin.js
    - dashboard.js
- utils
    - db.js
    - utility_functions.js

views

- partials
    - footer.ejs
    - header.ejs
    - nav_bar_student.ejs
    - nav_bar.ejs
    - student_script.ejs
    - task_mgmt_prof.ejs
    - task_mgmt_student.ejs
    - view_project.ejs
- admin.ejs
- home.ejs
- login.ejs
- mail_prof.ejs
- prof_conferences.ejs
- prof_students.ejs
- profs.ejs
- reading_list_prof.ejs
- reading_list_student.ejs
- register.ejs
- research_projects_prof.ejs
- research_projects_student.ejs
- sponsored_projects.ejs
- student_conferences.ejs
- under_review_papers_prof.ejs
- under_review_papers_student.ejs

.env
.gitignore
img_1.png
img_2.png
img_3.png
img_4.png
img.png
index.js
makefile
package-lock.json
package.json
README.md

Repository Structure

Root Directory

- `.idea/`
  Stores project metadata (used by IDEs like JetBrains products).
- `.gitignore`
  Specifies intentionally untracked files to ignore in version control (e.g., `.env`, node_modules).
- `modules.xml`, `Submission_Tracker.iml`, `vcs.xml`
  Configuration files for your IDE.
- `index.js`
  The entry point for the backend application. Configures middleware, routes, and starts the server.
- `makefile`
  Automation script to manage tasks like starting the server, running tests, etc.
- `package.json`
  Defines project dependencies, scripts, and metadata for the Node.js application.
- `package-lock.json`
  Tracks precise dependency versions for reproducible builds.
- `.env`
  Contains sensitive environment variables such as database credentials and server configurations.

Frontend

public/

- **prof_script.js**
  Handles client-side logic for professor-specific features.

views/

Contains EJS templates for server-side rendering.

- `partials/`
  - **`footer.ejs`**: Shared footer across pages.
  - **`header.ejs`**: Shared header with the site title and metadata.
  - **`nav_bar.ejs`**: Common navigation bar.
  - **`nav_bar_student.ejs`**: Navigation bar tailored for students.
- **Page Templates**
  - **`task_mgmt_prof.ejs`**: Page for professors to manage tasks.
  - **`task_mgmt_student.ejs`**: Page for students to manage tasks.
  - **`view_project.ejs`**: Displays project details.
  - **`admin.ejs`**: Admin dashboard for managing the system.
  - **`home.ejs`**: Landing page of the application.
  - **`login.ejs`**: Login page for authentication.
  - **`mail_prof.ejs`**: Mail interface for professors.
  - **`prof_conferences.ejs`**: Page to manage professor-related conferences.
  - **`prof_students.ejs`**: Page for professors to view and manage students.
  - **`profs.ejs`**: General professor-related information.
  - **`reading_list_prof.ejs`**: Reading list for professors.
  - **`reading_list_student.ejs`**: Reading list for students.
  - **`register.ejs`**: Registration page for new users.
  - **`research_projects_prof.ejs`**: Research projects dashboard for professors.
  - **`research_projects_student.ejs`**: Research projects dashboard for students.
  - **`sponsored_projects.ejs`**: Dashboard for managing sponsored projects.
  - **`student_conferences.ejs`**: Conference page for students.
  - **`under_review_papers_prof.ejs`**: Under-review papers for professors.
  - **`under_review_papers_student.ejs`**: Under-review papers for students.

Backend

src/

Contains server-side logic.

- **routes/**
  Organizes route files by user roles and functionalities.
  - prof_routes/
    - **conferences.js**: Handles professor-related conference routes.
    - **deadlines.js**: Manages professor deadlines.
    - **mail.js**: Handles professor email interactions.
    - **prof_dashboard.js**: Controls professor dashboard features.
    - **reading_list.js**: Reading list routes for professors.
    - **research_projects.js**: Routes for professor-specific research projects.
    - **sponsored_projects.js**: Manages sponsored projects.
    - **students.js**: Interacts with student data from the professor's view.
    - **todo.js**: Handles professor's to-do tasks.
    - **under_review_papers.js**: Manages papers under review for professors.
  - student_routes/
    - **conferences.js**: Routes for student-specific conferences.
    - **deadlines.js**: Manages student deadlines.
    - **profs.js**: Routes for student access to professor details.
    - **reading_list.js**: Reading list routes for students.
    - **research_projects.js**: Student-side research project routes.
    - **student_dashboard.js**: Manages the student dashboard.
    - **todo.js**: Handles students' to-do list tasks.
    - **under_review_papers.js**: Routes for students' under-review papers.
  - **Other Routes**

        - **admin.js**: Admin-specific routes for managing the system.
        - **dashboard.js**: Shared routes for dashboards.
- **utils/**
    - **db.js**: Sets up and manages database connections.
    - **utility_functions.js**: Contains helper functions used across the application.

Database

The backend for the **ReMa** (Research Management) system is built using PostgreSQL. It organizes data into multiple interconnected tables representing entities such as Professors, Students, Projects, Teams, Meetings, Conferences, Deadlines, and more. Below is a detailed explanation of the database schema.

SQL/

Contains database setup and population scripts.

- **Database_schemas.sql**: Defines the structure of all tables.
- **Data Population.sql**: Populates the database with initial data for testing and development.

Schemas

Database_schemas.sql

Schema Explanation

1. Professor Table

Definition: Manages data related to professors.
Columns:

- **id**: A unique identifier for each professor (Primary Key).
- **Name**: The name of the professor (up to 50 characters).
- **email_id**: The unique email address of the professor (ensures no duplicates).

2. Student Table

Definition: Manages data related to students.
Columns:

- **id**: A unique identifier for each student (Primary Key).
- **Name**: The name of the student (up to 50 characters).
- **email_id**: The unique email address of the student.
- **type**: Indicates the type of student (e.g., B.Tech, M.Tech) with a default value of 'B.Tech'.

---

3. Team Table

Definition: Links professors and students who are part of the same team.
Columns:

- **prof_id**: References the **Professor** table.
- **student_id**: References the **Student** table.
- **archived**: Marks whether the team is archived (default is FALSE).

---

4. Project Table

Definition: Stores information about research projects.
Columns:

- **id**: A unique identifier for each project (Primary Key).
- **title**: Title of the project (up to 1000 characters).
- **prof_table_id**: References the **Professor** table.
- **student_table_id**: References the **Student** table.
- **conference**: Conference associated with the project.
- **status**: Current status of the project.
- **link_1** & **link_2**: Links to project resources.
- **submitted_date** & **deadline_date**: Dates for submission and deadlines.
- **archived**: Marks if the project is archived.
- **sponsored**: Indicates if the project is sponsored.

- **paper**: Flags if the project includes a research paper.

---

5. Project_profs and Project_Students Tables

Definition: Tracks associations between projects and professors/students.
Columns:

- **project_id**: References the **Project** table.
- **prof_id**/**student_id**: References the **Professor**/**Student** table.
- **archived** (in Project_Students): Indicates if the student's involvement is archived.

Constraints: Foreign Key constraints enforce relationships between tables.

---

6. Meeting Notes Table

Definition: Logs meeting notes related to projects.
Columns:

- **id**: Unique identifier (Primary Key).
- **project_id**: References the **Project** table.
- **prof_id**: References the **Professor** table.
- **notes**: Stores the notes as text.
- **date**: Timestamp of the meeting (defaults to current timestamp).

---

7. Conferences Table

Definition: Records conferences associated with professors.
Columns:

- **prof_id**: References the **Professor** table.
- **id**: Unique identifier (Primary Key).
- **date**: Date of the conference.
- **title**: Title of the conference.

- **link**: Optional link to conference resources.

---

8. Deadlines and Todos Tables (for Professors and Students)

Definition: Tracks deadlines and to-do tasks for both professors and students.
Columns:

- **prof_id**/**student_id**: References the **Professor**/**Student** table.
- **id**: Unique identifier (Primary Key).
- **task**: Description of the deadline/task.
- **date**: Due date of the task.

---

9. Reading List Tables (for Professors and Students)

Definition: Stores recommended reading materials.
Columns:

- **id**: Unique identifier (Primary Key).
- **title**: Title of the material.
- **genre**: Genre of the material.
- **prof_id**/**student_id**: References the **Professor**/**Student** table.
- **conference**: Associated conference (if any).
- **status**: Status of the material (e.g., read, unread).
- **link_1** & **link_2**: Links to the material.

---

10. Admin Table

Definition: Stores administrator details.
Columns:

- **id**: Unique identifier (Primary Key).
- **Name**: Name of the admin.
- **email_id**: Unique email address of the admin.

Relationships and Constraints

- **Primary Keys**: Ensure the uniqueness of rows.
- **Foreign Keys**: Enforce relationships between entities.
- **ON DELETE CASCADE**: Automatically removes dependent records when a parent record is deleted, ensuring referential integrity.

The ReMa database schema is designed to effectively manage research-related activities, comprehensively tracking projects, teams, conferences, and deadlines. Its modular structure ensures scalability, maintainability, and data integrity, supporting the system's goals for efficient research management.

Assets

- **img_1.png, img_2.png, etc.**
  Image files for the application, possibly used in documentation or as part of the UI.