✅ Project Title:

ThesisHub: Automated Thesis Supervision Allocation & Management System

---

🔍 Problem Statement:

Currently, the thesis supervision process in our university's CSE department is handled manually. Students must form groups, submit preference lists of teachers, and wait for the chairman to manually sort and assign groups to supervisors based on CGPA and availability. This leads to inefficiency, delays, human error, and lack of transparency.

🎯 Objective:

To develop a web-based system that automates the thesis group formation, teacher preference submission, supervisor assignment, group-wise task distribution, and thesis evaluation process, ensuring fairness, transparency, and efficiency.

👨‍💻 Team Members:

- Abdullah Nazmus-Sakib (ID: 383) -> Functional Requirements
- Khaled Mahmud Jon (ID: 391) -> Non-Functional Requirements
- Shah Sultan (ID: 379) -> User story (student, teacher)
- Tawhidul Islam (ID: 395) -> User story of admin or chairman and rest of things

---

⚙️ Functional Requirements:

1. **User Roles:**
    - Student
    - Teacher
    - Chairman (Admin)
2. **Student Features:**
    - Register/Login
    - Form a group of two members
    - Submit teacher preference list
    - View assigned supervisor
    - View assigned tasks and progress status
    - Submit tasks for evaluation
    - View evaluation/marks
3. **Teacher Features:**

- Register/Login
- View group requests
- Accept up to 2 groups
- Create thesis tasks for assigned groups
- Evaluate task submissions
- Assign marks to students

4. **Chairman Features:**
   - Register/Login
   - View all student groups
   - Automatically sort groups by highest CGPA (top CGPA of group)
   - Automatically assign supervisors based on student preferences and teacher availability
   - Oversee the entire process
5. **Other Features:**
   - Group status dashboard
   - Task progress tracking
   - Notifications for all users
   - Mark sheet/report generation

🧪 Non-Functional Requirements:

1. **Usability:**

   The system should be intuitive and user-friendly for all roles.

2. **Performance:**

   The sorting and allocation algorithm should work efficiently even with 100+ groups.

3. **Security:**

   Role-based access control, password hashing, and secure data handling.

4. **Availability:**

   The system should be accessible 24/7 with minimal downtime.

5. **Maintainability:**

   The system should be built with modular code for future upgrades.

6. **Scalability:**

   Should support multiple departments in the future with minimal changes.

👥 User Stories:

1. Student:

I am a **student**,
so that I want to **form a group with my classmate**
so that **we can jointly apply for thesis supervision**.

I am a **student**,
so that I want to **submit a list of my preferred teachers**
so that **we have a chance to work with a teacher we admire**.

I am a **student**,
so that I want to **view the tasks assigned by our supervisor**
so that **we can work accordingly and submit them on time**.

I am a **student**,
so that I want to **view our group's evaluation and marks**
so that **I can understand how we are performing**.

---

2. Teacher:

I am a **teacher**,
so that I want to **see the groups assigned to me**
so that **I can provide them with necessary instructions**.

I am a **teacher**,
so that I want to **create specific thesis tasks and deadlines**
so that **students can stay on track**.

I am a **teacher**,
so that I want to **evaluate student progress and give marks**
so that **I can assess their thesis work fairly**.

---

3. Chairman (Admin):

I am the **chairman**,
so that I want to **view all student groups with their CGPA and choices**
so that **I can assign supervisors based on a fair and automated process**.

I am the **chairman**,
so that I want to **make sure each teacher only supervises up to 2 groups**
so that **workload is distributed evenly**.

I am the **chairman**,
so that I want to **oversee the entire supervision process from start to finish**
so that **no group is left unassigned or unattended**.

🛠️ Tools & Technologies (suggested):

- **Frontend:** HTML, CSS, JavaScript, Bootstrap/Tailwind
- **Backend:** Django (Python) or Laravel (PHP)
- **Database:** PostgreSQL or MySQL
- **Authentication:** Role-based login
- **Deployment:** Heroku / PythonAnywhere / Render / VPS

🗓️ Timeline (Rough):

Phase	Duration
Requirement Analysis	1 week
UI/UX Design	1 week
Backend + Database Design	1.5 weeks
Frontend Development	1.5 weeks
Integration & Testing	1 week
Documentation & Final Review	3-5 days