SOFTWARE REQUIREMENT SPECIFICATIONS (SRS) DOCUMENT

**NAME:** GOWTHAM K

**ROLL NO:** 7376221MC111

**YEAR:** II

**DEPARTMENT:** MECHATRONICS

**PROJECT ID:** 7

**PROJECT TITLE: “**TASK DASHBOARD ESSENTIAL FACULTY WORK STATUS PORTAL**”**

**TECH STACK:** MERN

# “TASK DASHBOARD - ESSENTIAL FACULTY WORK

# STATUS PORTAL”

### PROBLEM STATEMENT:

### Admin Dashboard Features:

The Admin dashboard uses JWT authentication for secure access. Admins can create, assign, edit, and delete tasks with details like title, description, due date, and assigned user. The dashboard shows task statuses (completed, active, inactive, overdue) and sends real-time notifications about upcoming deadlines and overdue tasks for proactive management.

### User Dashboard Features:

Users log in via secure JWT authentication to access their dashboard, viewing assigned tasks with details and deadlines. They can update task statuses (in-progress, completed). Real-time notifications for new tasks and upcoming deadlines are displayed, ensuring users manage their responsibilities effectively.

### Technology Stack and Implementation Plan:

The MERN stack is used: React.js for the frontend, Node.js with Express.js for the backend, and MongoDB for data storage. JWT ensures secure access.. The implementation involves setting up the project structure, developing authentication, building features, integrating notifications, and testing before deployment on platforms like Heroku or AWS.

### SOLUTION:

A web application called the "Task Dashborad - Essential faculty Work Status Portal." This portal will provide a centralized, real-time view of essential faculty work status, improving communication, efficiency, and responsiveness. This will address the limitations of the current system and create a more efficient and transparent work environment for both faculty and essential staff.

# SOFTWARE REQUIREMENT SPECIFICATIONS:

**INTRODUCTION:**

### PURPOSE:

The purpose of this project is to develop a web application that provides a central platform for tracking the work status of essential staff on campus. This will improve communication, efficiency, and transparency in managing maintenance tasks.

### PROJECT SCOPE:

The "Task Dashboard" web portal streamlines communication and workflow for managing essential staff. Faculty gain real-time insights into ongoing tasks, while supervisors update work status based on staff input. This collaborative system enhances efficiency and transparency in task maintenance.

### INDENTED AUDIENCE AND USE:

This portal is designed for two primary user groups:

1. **Faculty:** They will have access to a view-only dashboard displaying essential staff availability, and assigned tasks. This allows them to see their assigned task and track the progress of task requests.
2. **Supervisors:** They will have full access to the portal, including the ability to view and update staff information, task details, and work status. This allows them to monitor staff activity, manage task assignments, and ensure efficient completion of maintenance tasks

# SYSTEM WORKFLOW:

* 1. **Public Access (Faculty)**
  2. **Protected Access (Supervisor)**

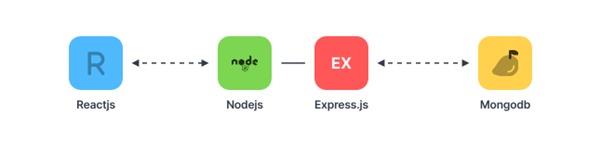
1. **Public Access (Faculty):**
   1. **Login:** Faculty users will log in with a secure username and password.
   2. **Dashboard:** Upon successful login, faculty will be directed to a view-only dashboard containing:
      1. **Upcoming task:** The number of task assigned for them on daily basis (e.g., Discipline duty or classes in the upcoming week).
      2. **Current Day’s Work:** A list of ongoing tasks for the current day, including:
         1. Location of the work
         2. Name of the work (brief description)
         3. Timing for the task (starting hr : min to ending hr : min)
         4. Overall status of the task (e.g., In Progress, On Hold, Completed)
      3. **Work History:** An archive of past maintenance tasks with details like those listed above.
      4. **View Staff Profile:** Faculty can view basic staff profiles containing information like name, contact details (if allowed), and area of expertise (if applicable).

#### Protected Access (Supervisors):

**Login** - **Supervisors will log in with a secure username and password. In addition to the functionalities available to faculty, supervisors will have the following privileges:**

1. **Create Staff:** Supervisors can add new essential staff members to the system, including their profile information.
2. **Create Work for the Current Day:** Supervisors can create new tasks for the current day, specifying:
   * Location of the work
   * Name of the work (detailed description)
   * Total number of Hours
   * Any additional details or instructions
3. **Update the Status of Current Day Works:** Supervisors can update the status of ongoing maintenance tasks throughout the day, reflecting progress or completion. This might involve:
   * Marking tasks as In Progress, On Hold, or Completed
   * Adding notes or remarks related to the task's progress

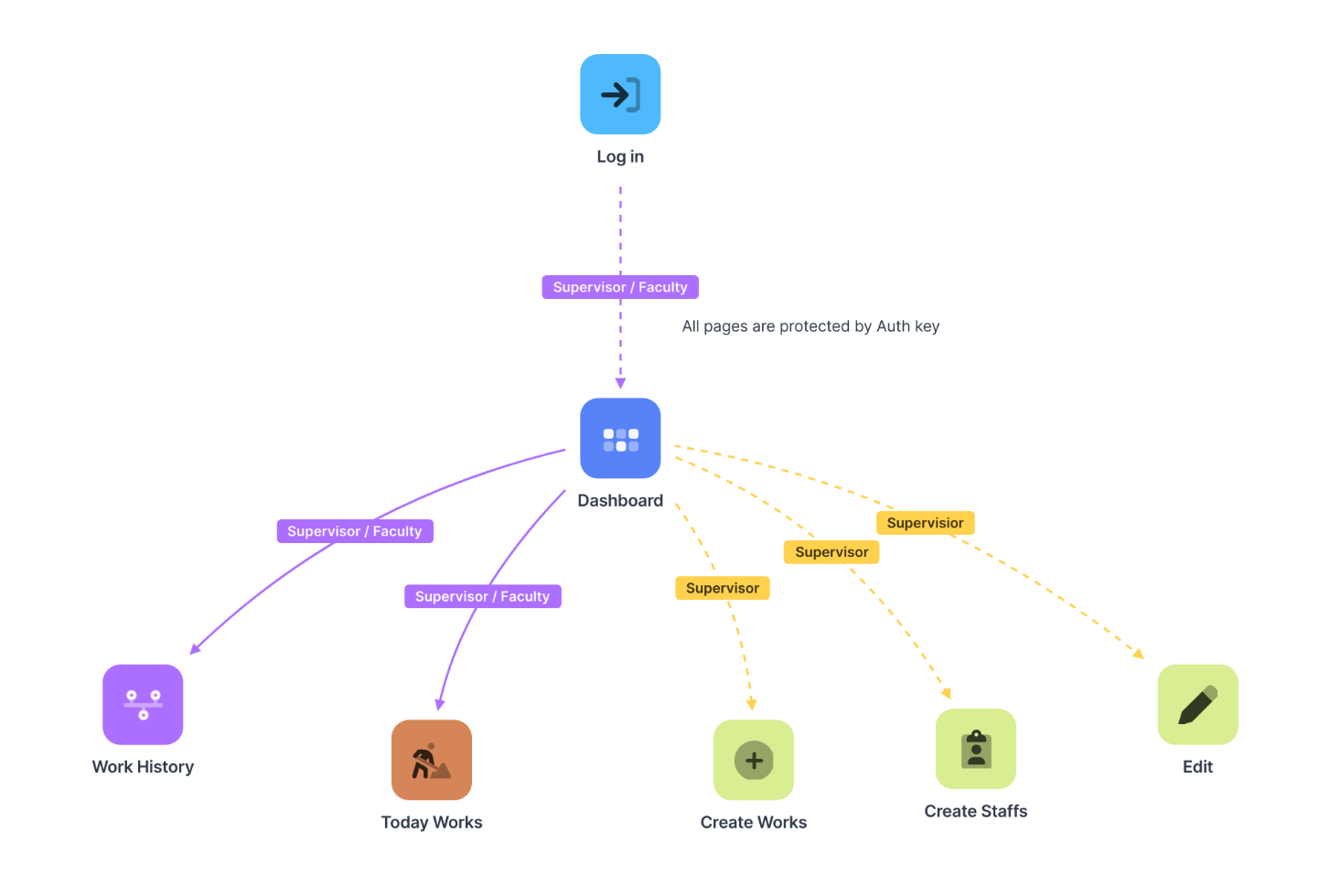
# TECHNOLOGY STACK:

**MERN – Mongodb ExpressJs ReactJs NodeJs**

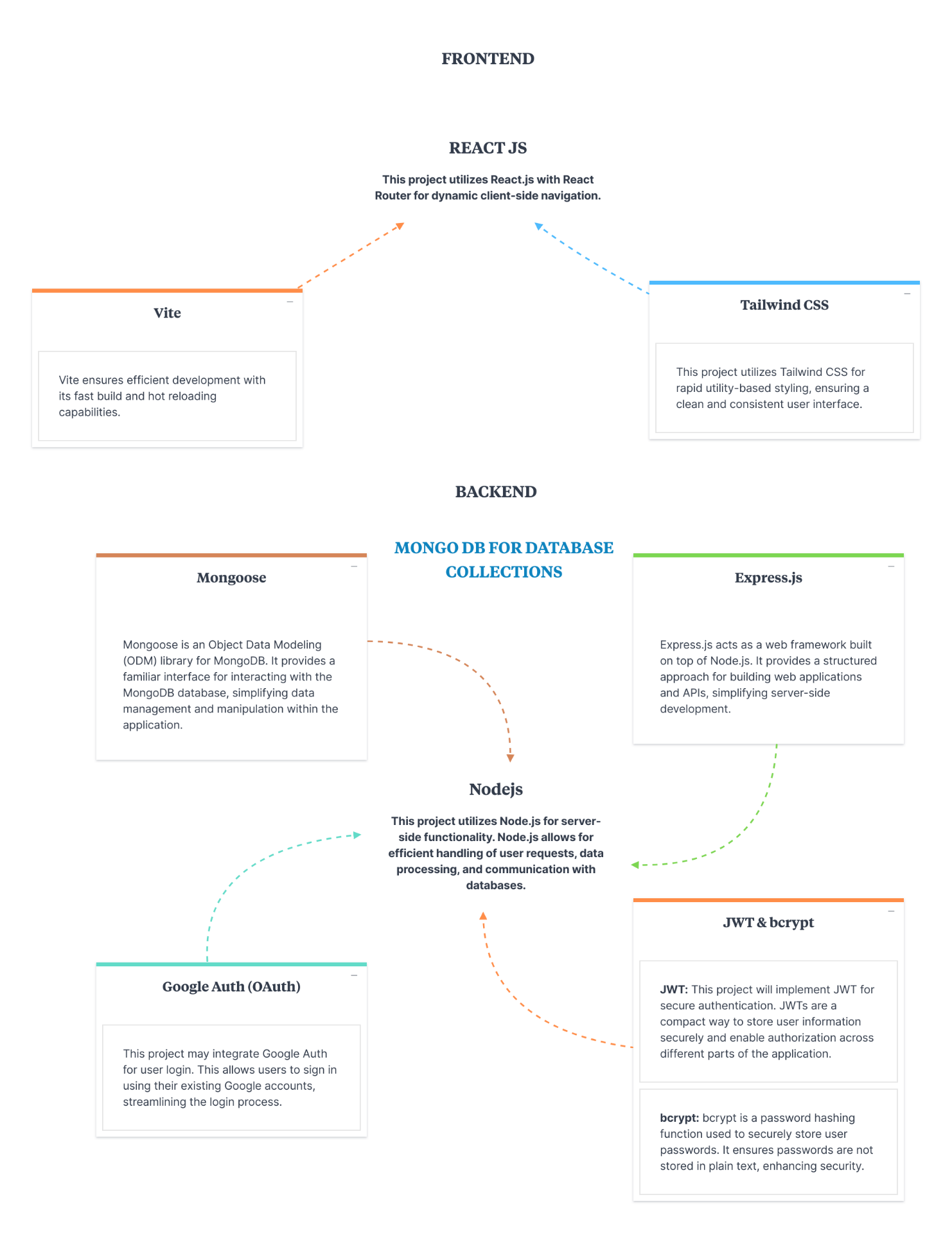
|  |  |
| --- | --- |
| **FRONTEND** | **REACT JS** |
| **BACKEND** | **NODE JS, EXPRESS JS** |
| **DATABASE** | **MONGODB** |
| **API** | **REST API** |

# SYSTEM FLOWCHART:

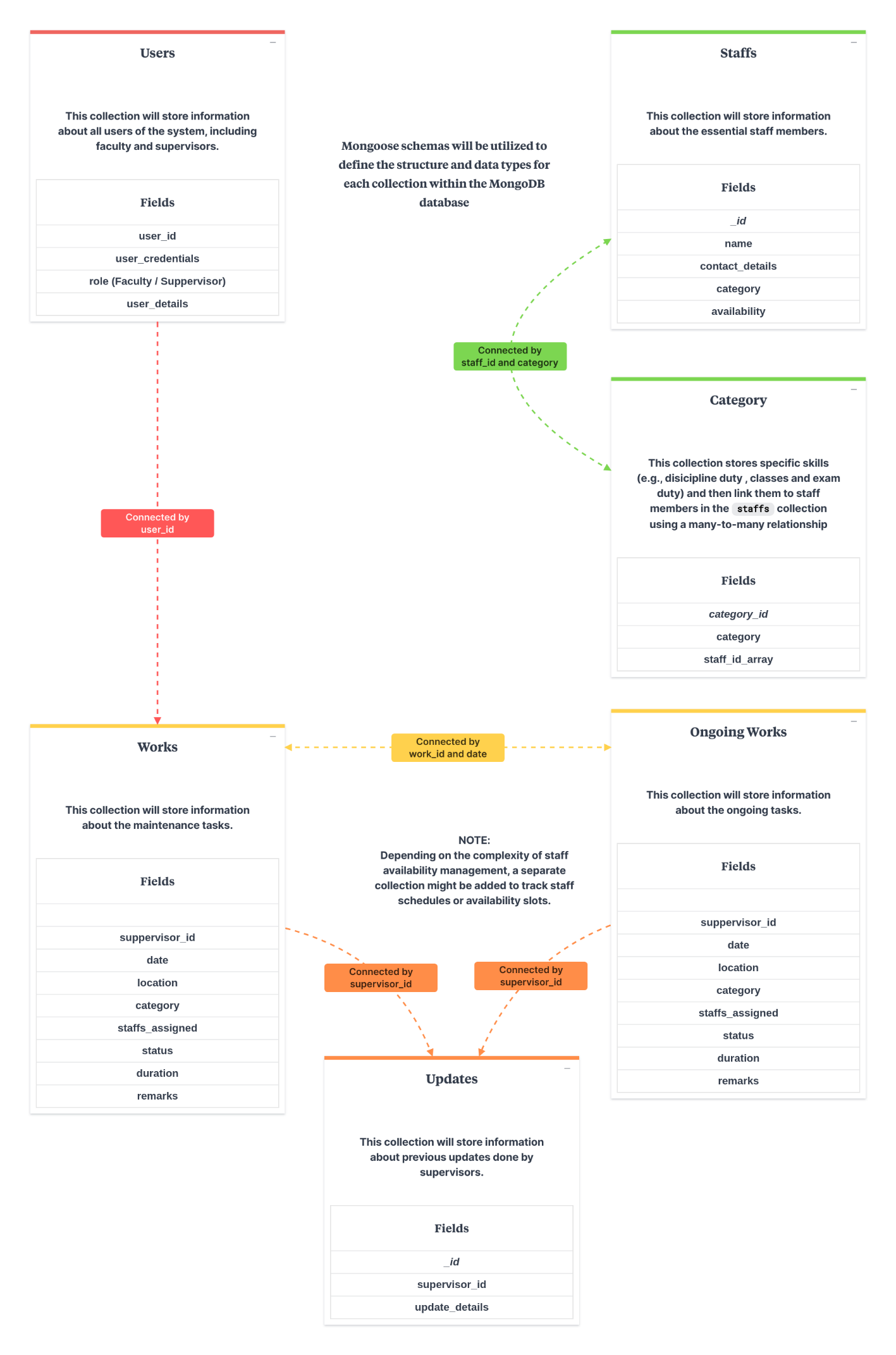
**OVERVIEW:**



**DEPENDENCIES:**



**DATABASE:**



# FUNCTIONAL REQUIREMENTS:

#### User Management:

1. **User Login:** All users (faculty and supervisors) must be able to log in using a secure username and password authentication system.
2. **User Roles:** The system might implement user roles (e.g., "faculty", "supervisor") to control access to specific functionalities (optional).

#### Staff Management (For Supervisors):

1. **Create Staff Profiles:** Supervisors can create profiles for essential staff members, including names, contact information, and category.
2. **Edit Staff Profiles:** Supervisors can edit existing staff profiles as needed (e.g., update contact details, availability).

#### Work Management:

1. **Create Work Orders:** Supervisors can create new work orders for the current day, specifying:
   1. Location of the work (block)
   2. Nature of the work (detailed description of the task)
   3. Timing of the Task
   4. Any additional details or instructions
2. **Update Work Status:** Supervisors can update the status of ongoing tasks throughout the day, reflecting progress or completion. This might involve:
   1. Marking tasks as In Progress, On Hold, or Completed
   2. Adding notes or remarks related to the task's progress

#### Faculty Access:

1. **View Work Status:** Faculty will have access to a view-only dashboard displaying real-time information on essential staff work status. This might include:
   1. Location of ongoing tasks
   2. Nature of the work (brief description)
   3. Staff members assigned (names or number)
   4. Overall status of the task (e.g., In Progress, On Hold, Completed)
2. **View Staff Profiles:** Faculty may have access to basic staff profiles containing information like name, contact details, and area of expertise.

#### Security:

1. Role-based access control (RBAC) can be implemented to restrict access to certain functionalities based on user roles.
2. Secure data storage practices should be followed to protect sensitive information like staff contact details.

# EXTERNAL REQUIREMENTS:

## User Interface (UI):

1. **Accessibility:** The UI should comply with WCAG (Web Content Accessibility Guidelines) to ensure usability for users with disabilities. This might involve features like:
2. **Responsiveness:** The UI should display and function correctly across various devices with different screen sizes (e.g., desktops, laptops, tablets).
3. **Browser Compatibility:** The web application should function properly on all major web browsers (e.g., Chrome, Firefox, Safari) on various operating systems (e.g., Windows, macOS).

## Software Interface (SI):

1. **Security:** The system should integrate with a secure user authentication system for logins (e.g., existing college authentication).
2. **Data Integration:** The system may need to integrate with existing databases to retrieve staff information or building details.
3. **API Usage:** If the system exposes data or functionalities through APIs for external applications, these APIs should be secure. This includes authentication, authorization, and data validation mechanisms.

## NON-FUNCTIONAL REQUIREMENTS (NFRs):

#### Performance:

* 1. **Concurrent Users:** Specify the expected number of concurrent users the system can support.
  2. **Scalability:** The system should be scalable to accommodate future growth in users and tasks.

#### Security:

1. **Authentication And Authorization:** Implement secure user authentication with username and password hashing to protect user credentials. Role-based access control (RBAC) to restrict access to functionalities based on user roles (e.g., faculty can only view, supervisors can create and edit).
2. **Data Security:** Encrypt sensitive information like staff contact details at rest and in transit. Implement access controls to restrict unauthorized access to sensitive data.

#### Compatibility:

1. **Device And Browser Compatibility:** The web application should function correctly across all major web browsers (e.g., Chrome, Firefox, Safari). Consider accessibility on different devices (e.g., desktops, laptops, tablets) if relevant for the user base.

#### Reliability and Availability:

1. **Uptime:** Define the target uptime percentage for the system (e.g., 99.7%). This reflects how often the system is expected to be available for users.

#### Usability:

1. **Intuitive Interface:** The user interface should be intuitive and easy to navigate for both faculty and supervisors.
2. **User Training:** Consider providing basic user training materials (e.g., user guides, video tutorials) to help faculty and supervisors understand how to use the system effectively.

#### Maintainability:

1. **Error Handling:** Implement robust error handling mechanisms to gracefully handle unexpected situations and provide informative error messages to users.
2. **Version Control:** Use a version control system (Git) to track code changes and facilitate collaboration among developers.