Software Requirements Specification

Faculty Project Matching Portal

Version 1.0

Sri Spoorthi Vattem (SE22UARI165)
A.A. Vivek Showry (SE22UARI014)
Sidhanth Ranjan (SE22UARI161)
Adya Bahl (SE22UARI205)
Reddyshetty Kruthi (SE22UARI142)

Instructor: Dr. Arun Avinash Chauhan

Course: Software Engineering

Lab Section: AI-3 & AI-1

Teaching Assistant: Murali Krishna Bukkasamudram

Date: 19th February 2025

Table of Contents

1. Introduction

1.1 Document Purpose

This document provides a comprehensive specification for the Faculty Project Matching Portal, which is designed to facilitate project selection, application, and approval processes for final-year students at Mahindra University. This system will enable faculty to list projects and students to apply efficiently.

1.2 Product Scope

The Faculty Project Matching Portal is a web-based application that streamlines the process of students applying for faculty-led projects. Faculty members can list their available projects, review applications, and approve/reject students. Additionally, the system will provide basic scheduling for meetings outside the platform.

1.3 Intended Audience and Document Overview

This document is intended for software developers, faculty members, students, and university administrators involved in the development and deployment of the system. It outlines the functional and non-functional requirements, system architecture, and constraints.

1.4 Definitions, Acronyms, and Abbreviations

- MU: Mahindra University
- UI: User Interface
- SQLite: Lightweight database engine
- Flask/Django: Web development frameworks

2. Overall Description

2.1 Product Overview

The Faculty Project Matching Portal is an independent web application designed to help students find and apply for faculty projects. The system will store project details, manage applications, and facilitate communication between students and faculty.

2.2 Product Functionality

The system will include the following major functionalities:

- Faculty can list projects.
- Students can browse and apply for projects.
- Faculty can approve/reject applications.
- Students can track their application status.
- Basic meeting scheduling between students and faculty.

2.3 Design and Implementation Constraints

The system will be hosted on a university server and use SQLite as the primary database. The development framework will be Flask or Django.

2.4 Assumptions and Dependencies

- The system will be used internally within the university.
- Security measures will be minimal due to controlled access.
- The scheduling feature will be limited to basic time-slot management.

3. Specific Requirements

3.1 External Interface Requirements

User Interfaces: Simple web-based UI using HTML and CSS.

Hardware Interfaces: Hosted on a university-provided server.

Software Interfaces: Uses Flask/Django for backend processing and SQLite for data storage.

3.2 Functional Requirements

F1: Faculty should be able to list new projects.
F2: Students should be able to browse available projects.
F3: Students should be able to submit applications.
F4: Faculty should be able to approve/reject applications.
F5: The system should allow students and faculty to track application status.
F6: The system should include a basic meeting scheduling feature.

4. Other Non-functional Requirements

4.1 Performance Requirements

- The system should support up to 100 concurrent users.
- The response time for queries should not exceed 7 seconds.

4.2 Safety and Security Requirements

- The system should restrict access to authorized university users.
- No sensitive student data should be stored beyond application details.

4.3 Software Quality Attributes

- Reliability: The system should have an uptime of at least 99%.
- Usability: The UI should be simple and easy to navigate.
- Maintainability: The codebase should be modular for easy updates.