Functional Specification Document (FSD) Template

1. Introduction

This Functional Specification Document (FSD) outlines the detailed functional requirements for the [Project Name] system. It serves as a comprehensive guide for development, testing, and stakeholder communication, ensuring a shared understanding of the system's intended behavior and capabilities. This document will describe the system's features, user interactions, data flows, and expected outcomes.

• **Project Title**: Hematovision: advanced blood cell classification using transfer learning

Team ID : LTVIP2025TMIDS66117

• Team Members:

Team Size : 4

Team Leader : Kurra Madhu Priya

Team member : Kurapati Sri Lakshmi

Team member : Chaitanya Nandini Nimmala

Team member : Munagala Madhumitha Kumari

1.1 Purpose of this Document

The primary purpose of this FSD is to:

- Define the functional requirements of the system in a clear, unambiguous, and verifiable manner.
- Serve as a reference for developers during implementation.
- Provide a basis for quality assurance and testing activities.
- Facilitate communication and agreement among all project stakeholders.

1.2 Scope of the System

[Briefly describe what the system will and will not do. Define the boundaries of the project.]

1.3 Target Audience

This document is intended for project managers, business analysts, developers, quality assurance engineers, and other relevant stakeholders involved in the [Project Name] project.

1.4 Definitions, Acronyms, and Abbreviations

[Provide a glossary of terms, acronyms, and abbreviations used throughout the document.]

2. Overall Description

2.1 Product Perspective

[Describe how the system fits into the larger context or ecosystem. Is it a standalone application, part of a suite, or an enhancement to an existing system?]

2.2 Product Functions

[Provide a high-level summary of the major functions the system will perform. This section should give a general understanding of the system's capabilities without going into excessive detail.]

2.3 User Characteristics

[Describe the different types of users who will interact with the system and their relevant characteristics (e.g., technical proficiency, roles, responsibilities).]

2.4 General Constraints

[List any general constraints that will affect the design and development of the system, such as regulatory requirements, hardware limitations, software dependencies, operational environment, or performance requirements.]

3. Functional Requirements

This section details the specific functional requirements of the system, organized by feature or module. Each requirement should be clear, concise, and testable.

3.1 [Feature/Module Name 1]

3.1.1 [Functional Requirement 1.1]

[Detailed description of the functional requirement. What action does the system perform? What are the inputs? What are the outputs? What are the pre-conditions and postconditions?]

- **Description:** [Elaborate on the requirement.]
- **Inputs:** [List inputs required for this function.]
- **Outputs:** [List outputs generated by this function.]
- **Pre-conditions:** [Conditions that must be true before the function can be executed.]
- **Post-conditions:** [Conditions that will be true after the function has been successfully executed.]
- **Business Rules:** [Any specific business rules associated with this function.]
- **Error Handling:** [How the system should behave in case of errors related to this function.]

3.1.2 [Functional Requirement 1.2]

[Repeat the structure for each functional requirement within this feature/module.]

3.2 [Feature/Module Name 2]

[Repeat the structure for other features/modules.]

4. Non-Functional Requirements

This section describes the non-functional requirements, which specify criteria that can be used to judge the operation of a system, rather than specific behaviors.

4.1 Performance Requirements

[e.g., response times, throughput, capacity, scalability.]

4.2 Security Requirements

[e.g., authentication, authorization, data encryption, access control.]

4.3 Usability Requirements

[e.g., ease of use, learnability, user interface standards.]

4.4 Reliability Requirements

[e.g., availability, fault tolerance, recovery from failures.]

4.5 Maintainability Requirements

[e.g., ease of modification, testability, code standards.]

4.6 Portability Requirements

[e.g., operating system compatibility, browser compatibility.]

5. Data Model (Optional)

[Describe the data entities, their attributes, relationships, and constraints. This can include entity-relationship diagrams (ERDs) or data dictionaries.]

6. User Interface Requirements (Optional)

[Describe the user interface elements, screen layouts, navigation, and interaction patterns. This can include wireframes, mockups, or prototypes.]