PES UNIVERSITY

Department of Computer Science and Engineering

UE22CS341A: Software Engineering

PROJECT GUIDELINES

Acknowledgment: Authors of the document

Purpose: As part of the software engineering course, students are expected to do a project incorporating all stages of software development life cycle. As the units will be taught in the classes, phase wise inclusion needs to be done.

Outcomes: Team Details, Project Title, Synopsis, all intermediate SE(SDLC) Documents and a complete Project Report (Formats attached for reference)**.**

Evaluation and Complete Process: The details of how this project must be done and evaluated are as described below:

Guidelines:

1: Project team must identify an application case study (DBMS Project)

2: Team must prepare a synopsis to enlist all the features of the chosen project / software application.

3: Prepare a document consisting of requirements, planning, design, implementation, and testing details.

4: Project Report consisting of all the relevant documents must be submitted with all results.

Project Team:

The project team will be done as per the guidelines of DBMS Project. Evaluations will be done based on individual performance and contribution made towards the project. During said time, the document submission is mandatory. This will carry a portion of the total marks awarded for the project.

PES UNIVERSITY
Department of Computer Science and Engineering
UE22CS341A: Software Engineering

Deliverable 1: Synopsis and SRS (To submit on *Aug 31, 2024)*

Software Requirements Specification

for

<Project Title>

Version 1.0 approved

Prepared by <author>

<Organization>

<Date created>

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

PES UNIVERSITY

BENGALURU

Department of Computer Science and Engineering

Introduction

Purpose

<Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by this SRS, particularly if this SRS describes only part of the system or a single subsystem.>

Intended Audience

<Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. Describe what the rest of this SRS contains and how it is organized.

Product Scope

<Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals. Relate the software to corporate goals or business strategies. If a separate vision and scope document is available, refer to it rather than duplicating its contents here.>

References

<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>

Overall Description

Product Perspective

<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>

Product Functions

<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or object class diagram, is often effective.>

User Classes and Characteristics

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>

Operating Environment

<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>

Design and Implementation Constraints

<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer's organization will be responsible for maintaining the delivered software).>

2.6 Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

External Interface Requirements

User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

<if any Hardware Interfaces include as 3.4 Hardware Interfaces>

Analysis Models

<Include pertinent analysis models, such as use case diagrams and if applicable entity-relationship diagrams.>

System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

System Feature 1

<Don't really say "System Feature 1." State the feature name in just a few words.>

5.1.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

5.1.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

5.1.3 Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use "TBD" as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1:
REQ-2:

System Feature 2 (and so on)

Other Non Functional Requirements

Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be

taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product's design or use. Define any safety certifications that must be satisfied.>

Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

Also include Domain requirements here.

Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Field Layouts

An Excel sheet containing field layouts and properties/attributes and report requirements.

Sample sheet with information required to register the customer

Field	Length	Data Type	Description	Is Mandatory
Account Number	16	Numeric		Y
ISFC code	11	Alphanumeric		Y
Card Amount	20	Numeric		Y
Mandate Start Date	8	Date	Date of Mandate Registration	N
Mandate End Date	8	Date	Date of Mandate Expiry	N
Status	25	Alphanumeric	Status of Registration	
Y Customer Name	60	String		Y
Reject Reason Code	4	String	Reject Reason code in case mandate is rejected	N

Sample Report Requirements: Include the fields to be included in the

report Registration Report	Transaction Report
Bank Account Number	Transaction Reference
Number ISFC Code	Bank Account Number
Bank Name	IFSC Code
Account Status	Bank Name

Account Type	Customer Name
Customer Name	Card Number
Card Number	Debit Transaction Amount
SI Start Date	Transaction Date
Status	Status
Remarks	Debit Attempt Number Remarks

Appendix C: Requirement Traceability Matrix

Sl. No	Requirement ID	Brief Description of Requirement	Architecture Reference	Design Reference	Code File Reference	Test Case ID	System Test Case ID

PES UNIVERSITY

BENGALURU

Department of Computer Science and Engineering

Deliverable 2: Planning, designing and implementation details(To submit on Sep 30, 2024)

Project Plan Document

Instructions:

Prepare a detailed plan for your project which comprises of the below mentioned

details. Upload pdf document through the given link.

The name of the document should be your Project ID.

Things to be included as part of the project plan.

1. Identify the lifecycle to be followed for the execution of your project and justify why you have chosen the model.
2. Identify the tools which u want to use it throughout the lifecycle like planning tool, design tool, version control, development tool, bug tracking, testing tool.
3. Determine all the deliverables and categorize them as reuse/build components and justify the same.
4. Create a WBS for the entire functionalities in detail.
5. Do a rough estimate of effort required to accomplish each task in terms of person

   months. Create the Gantt Chart for scheduling using any tool

6. Coding Details

PES UNIVERSITY

BENGALURU

Department of Computer Science and Engineering

Deliverable 3: Testing details (to submit on Oct 20, 2024)

Test Plan Document

Instructions:

1: Prepare at least 8-10 test cases for each **implemented use case** (functional requirements) as per the below given template which should include Unit, Integration and System Test cases.

2: Carry out manual testing for all the test cases and populate the columns Actual Result and Test Result.

Template of a Test Case:

Test Case ID	Name of Module	Test Case Description	Pre-conditions	Test Steps	Test Data	Expected Results	Actual Result	Test Result

- **Test Case ID :** Each test case should be represented by a unique ID. To indicate test types, follow some convention like "UT_01" indicating "Unit Testing - Test Case#1."
- **Name of the Module :** Specify the name of the **main module or sub module** being tested
- **Test Case Description :** Specify the summary or test purpose in brief
- **Pre- Conditions :** Any requirement that needs to be done before execution of this test case.
- **Test Steps** : Mention all the steps in detail and specify the order in which it is to be executed.
- **Test Data :** Input for the test case to be executed. Specify different data sets with precise values to be used as input. (create test case for both valid and invalid inputs)
- **Expected Results :** Mention the expected results including error or precise messages that should be displayed on screen
- **Actual Results :** After execution of test case fill this column with the result obtained
- **Test Result (Pass/Fail) :** Mark this field as “fail” if the actual result is not same as expected result else mark as “pass”.

PES UNIVERSITY

BENGALURU

Department of Computer Science and Engineering

Sample Test Case:

Test Case ID	Name of Module	Test case description	Pre-conditions	Test Steps	Test data	Expected Results	Actual Result	Test Result
UT-01	User registration module	To test the login functionality	Access to Chrome Browser	1: Navigate to http://www.demo.com 2: Enter Username and Password 3: Click Submit	User name: PESU Student. Password : pes123	Login should be successful with "welcome message"	Login successful with "welcome message" displayed	Pass

PES UNIVERSITY

BENGALURU

Department of Computer Science and Engineering

Deliverable 4: Final Report (To submit on Nov 12, 2024)

Department of Computer Science and Engineering

BENGALURU, KARNATAKA, INDIA.
B. TECH. (CSE) V

SEMESTER

Aug. – Dec. 2024

UE22CS341A – SOFTWARE ENGINEERING PROJECT REPORT

ON

<PROJECT TITLE>

SUBMITTED BY

SRN -- NAME

1) **Student Name**
2) **Student Name**
3) **Student Name**

PES UNIVERSITY

BENGALURU

Department of Computer Science and Engineering

TABLE OF CONTENTS		
Sl. No.	TOPIC	PAGE No
1.	Proposal Of The Project	
2.	Software Requirements Specification	
3.	Project Plan	
4.	Design Diagrams	
5.	Test Cases	
6.	Results (Screen Shots)	