

# Project Synopsis

For

**An AI-Driven System for Automated Requirements  
Engineering from  
Unstructured Stakeholder Communication**

**07/02/2026**

Prepared by

<b>Specialization</b>	<b>SAP ID</b>	<b>Name</b>
DevOps	500122740	Shreyash Shivhare
DevOps	500123425	Sreyas Sharma
DevOps	500123468	Devanshi Jain
DevOps	500124202	Sreyas Sharma

School Of Computer Science  
UNIVERSITY OF PETROLEUM & ENERGY STUDIES,  
DEHRADUN- 248007. Uttarakhand

# Table of Contents

Topic		Page No
Table of Content		
Revision History		
1	Introduction	
	1.1 Purpose of the Project	
	1.2 Target Beneficiary	
	1.3 Project Scope	
	1.4 References	
2	Project Description	
	2.1 Reference Algorithm	
	2.2 Data/ Data structure	
	2.3 SWOT Analysis	
	2.4 Project Features	
	2.5 User Classes and Characteristics	
	2.6 Design and Implementation Constraints	
	2.7 Design diagrams	
	2.8 Assumption and Dependencies	
3	System Requirements	
	3.1 User Interface	
	3.2 Software Interface	
	3.3 Database Interface	
	3.4 Protocols	
4	Non-functional Requirements	
	4.1 Performance requirements	
	4.2 Security requirements	
	4.3 Software Quality Attributes	
5	Other Requirements	
Appendix A: Glossary		
Appendix B: Analysis Model		
Appendix C: Issues List		

1	INTRODUCTION	
	1.1 Purpose of the Project	The purpose of this project is to develop an <b>AI-Driven Automated Requirements Engineering System</b> that automatically extracts, classifies, clusters, prioritizes, and summarizes software requirements from natural language inputs such as stakeholder interviews, emails, and requirement documents. Traditional requirements engineering processes are manual, time-consuming, and prone to ambiguity and errors. This system aims to reduce human effort, improve requirement accuracy, and accelerate the software development lifecycle
	1.2 Target Beneficiary	<ul style="list-style-type: none"> <li>• Software development teams</li> <li>• Requirement analysts and system analysts</li> <li>• Project managers</li> <li>• Software organizations and startups</li> <li>• Academic researchers working in Requirements Engineering and NLP</li> </ul>
	1.3 Project Scope	<p>The proposed system operates in the software development and requirements engineering domain. It processes natural language requirement documents and generates structured outputs such as classified requirements (functional/non-functional), clustered requirement groups, prioritized lists, and summarized reports.</p> <p><b>Objectives:</b></p> <ul style="list-style-type: none"> <li>• Automate requirement extraction from text</li> <li>• Classify requirements using machine learning</li> <li>• Cluster similar requirements for organization</li> <li>• Prioritize requirements based on importance</li> <li>• Generate structured summarized reports</li> </ul> <p><b>Deliverables:</b></p> <ul style="list-style-type: none"> <li>• NLP-based requirement processing model</li> <li>• Basic web-based prototype interface</li> <li>• Preprocessed datasets and trained models</li> <li>• Final project report and demonstration</li> </ul>
	1.4 References	<p>[1] K. Lam, M. H. Bhuiyan, M. S. Islam, A. H. Chowdhury, Z. A. Bhuiyan, and S. Ahmmed, "Co-pilot for project managers: Developing a PDF-driven AI chatbot for facilitating project management," IEEE Access, vol. 13, pp. 43079–43096, 2025, doi: 10.1109/ACCESS.2025.3548519.</p> <p>[2] V. Siddeshwar, S. Alwidian, and M. Makrehchi, "A systematic review of AI-enabled frameworks in requirements elicitation," IEEE Access, vol. 12, pp. 154310–154328, 2024, doi: 10.1109/ACCESS.2024.3475293.</p> <p>[3] M. A. Umar and K. Lano, "Advances in automated support for requirements engineering: A systematic literature review,"</p>

		<p>Requirements Engineering, vol. 29, pp. 177–207, 2024, doi: 10.1007/s00766-023-00411-0.</p> <p>[4] G. Voria, F. Casillo, C. Gravino, G. Catolino, and F. Palomba, “RECOVER: Toward requirements generation from stakeholders’ conversations,” IEEE Transactions on Software Engineering, vol. 51, no. 6, pp. 1912–1927, 2025, doi: 10.1109/TSE.2025.3572056.</p>
2	<b>PROJECT DESCRIPTION</b>	
	2.1 Reference Algorithm	<p>The system follows an <b>NLP-based requirements processing pipeline</b> consisting of the following steps:</p> <ol style="list-style-type: none"> <li>1. Input requirement text collection</li> <li>2. Text preprocessing (tokenization, stop-word removal, stemming/lemmatization)</li> <li>3. Requirement extraction using NLP techniques</li> <li>4. Requirement classification using Machine Learning models</li> <li>5. Requirement clustering using similarity-based clustering algorithms</li> <li>6. Requirement prioritization using scoring techniques</li> <li>7. Requirement summarization and structured output generation</li> </ol> <p><b>Required Data Structures:</b></p> <ul style="list-style-type: none"> <li>• Text datasets stored as structured CSV/JSON</li> <li>• Vectorized representations using TF-IDF matrices</li> <li>• Requirement clusters stored using array/list structures</li> </ul>
	2.2 Characteristic of Data	<p>The dataset used in the project consists of <b>natural language requirement documents</b>, including software requirement specifications, stakeholder requirement statements, and publicly available requirement datasets.</p> <p><b>Primary Data Sources:</b></p> <ul style="list-style-type: none"> <li>• Public requirements datasets (PURE dataset, Kaggle SRS datasets)</li> <li>• Sample requirement documents prepared for experimentation</li> </ul> <p><b>Secondary Data Sources:</b></p> <ul style="list-style-type: none"> <li>• Research paper datasets</li> <li>• Open-source software requirement repositories</li> </ul> <p><b>Sampling Technique:</b></p> <p>Random sampling of requirement sentences from collected datasets.</p> <p><b>Statistical / Processing Methods:</b></p> <ul style="list-style-type: none"> <li>• Text preprocessing using NLP techniques</li> <li>• Feature extraction using TF-IDF</li> <li>• Machine learning classification models</li> <li>• Clustering algorithms for grouping similar requirements</li> </ul>

	2.3 SWOT Analysis	<p><b>Strengths</b></p> <ul style="list-style-type: none"> <li>Automates manual requirements engineering tasks using AI and NLP</li> <li>Reduces time, human effort, and requirement ambiguity</li> <li>Improves requirement organization and documentation quality</li> <li></li> </ul> <p><b>Weaknesses</b></p> <ul style="list-style-type: none"> <li>Accuracy depends on dataset quality and model training</li> <li>Requires preprocessing and computational resources</li> </ul> <p><b>Opportunities</b></p> <ul style="list-style-type: none"> <li>Can be integrated into software development lifecycle tools</li> <li>Useful for startups and organizations managing large requirement datasets</li> </ul> <p><b>Threats</b></p> <ul style="list-style-type: none"> <li>Rapid changes in NLP technologies</li> <li>Availability of commercial requirement automation tools</li> </ul>
	2.4 Project Features	<ul style="list-style-type: none"> <li>Upload or input natural language requirement documents</li> <li>Automatic requirement extraction and preprocessing</li> <li>Classification of requirements (functional/non-functional)</li> <li>Clustering of similar requirements</li> <li>Requirement prioritization based on importance</li> <li>Generation of summarized structured requirement reports</li> <li>Basic web interface for user interaction</li> </ul>
	2.5 User Classes and Characteristics	<p><b>Requirement Analysts:</b></p> <p>Use the system to process requirement documents and generate structured outputs.</p> <p><b>Software Developers:</b></p> <p>Use structured requirement reports for system development planning.</p> <p><b>Project Managers:</b></p> <p>Use prioritized requirement lists for project planning and decision-making.</p> <p><b>Researchers/Students:</b></p> <p>Use the system for experimentation and academic research.</p>

	2.6 Design and Implementation Constraints	<ul style="list-style-type: none"> <li>The system will run on standard computing devices with moderate memory requirements.</li> <li>Implementation will use Python, NLP libraries (NLTK/spaCy), and machine learning frameworks.</li> <li>A web-based interface will be developed for accessibility.</li> <li>The system requires internet access for deployment and dataset retrieval (if applicable).</li> <li>Security considerations include safe handling of uploaded requirement documents and restricted user access.</li> </ul>
	2.7 Design diagrams	<p>The following design diagrams will be prepared for the system:</p> <ul style="list-style-type: none"> <li>Use Case Diagram showing user interactions with the requirement processing system</li> <li>Class Diagram representing system classes such as Requirement, User, Processing Module, and Report Generator</li> <li>Activity Diagram describing the workflow of requirement processing</li> <li>Sequence Diagram illustrating interaction between system modules</li> <li>Data Flow Diagram (DFD) representing input requirement flow and processed outputs</li> <li>State Diagram representing requirement processing states</li> </ul>
	2.8 Assumption and Dependencies	<p><b>Assumptions</b></p> <ul style="list-style-type: none"> <li>Users will provide requirement documents in readable text format.</li> <li>Sufficient dataset availability for training and testing machine learning models.</li> <li>Users will have internet-enabled systems to access the web interface.</li> </ul> <p><b>Dependencies</b></p> <ul style="list-style-type: none"> <li>NLP libraries such as NLTK, spaCy, or Scikit-learn</li> <li>Availability of training datasets for requirement classification</li> <li>Hosting environment for deployment of the web interface</li> </ul>
3	<b>SYSTEM REQUIREMENTS</b>	
	3.1 User Interface	<p>A web-based user interface will allow users to:</p> <ul style="list-style-type: none"> <li>Upload requirement documents</li> <li>View processed classified and clustered requirements</li> <li>Access summarized requirement reports</li> <li>Download structured outputs</li> </ul>
	3.2 Software Interface	<p>The system modules include:</p> <ul style="list-style-type: none"> <li>Input module for requirement upload</li> <li>NLP processing module for text preprocessing</li> <li>Machine learning module for classification and clustering</li> </ul>

		<ul style="list-style-type: none"> <li>Reporting module for generating summarized outputs</li> </ul> <p>Communication between modules will occur through internal APIs implemented using Python-based frameworks.</p>
	3.3 Database Interface	<p>The system will use a lightweight database (SQLite / MongoDB) to store:</p> <ul style="list-style-type: none"> <li>Uploaded requirement documents</li> <li>Processed requirement outputs</li> <li>Model results and logs</li> </ul>
	3.4 Protocols	<ul style="list-style-type: none"> <li>HTTP/HTTPS protocol for web communication</li> <li>Secure file upload handling</li> <li>Standard REST-based communication between frontend and backend modules</li> </ul>
4	NON-FUNCTIONAL REQUIREMENTS	
	4.1 Performance requirements	<ul style="list-style-type: none"> <li>The system should process uploaded requirement documents within a reasonable processing time depending on document size.</li> <li>The application should support multiple document uploads without system failure.</li> <li>Response time for report generation should remain efficient under moderate usage conditions.</li> </ul>
	4.2 Security requirements	<ul style="list-style-type: none"> <li>User-uploaded requirement documents must be securely stored and accessed only by authorized users.</li> <li>Secure communication should be maintained using HTTPS.</li> <li>Access control mechanisms should restrict unauthorized system usage.</li> </ul>
	4.3 Software Quality Attributes	<p><b>Adaptability:</b></p> <p>The system can be updated to support new datasets, algorithms, or requirement formats with minimal modification.</p> <p><b>Availability:</b></p> <p>The system will be available through a web-based interface accessible to authorized users.</p> <p><b>Correctness:</b></p> <p>Machine learning and NLP models will ensure accurate extraction, classification, and summarization of requirements.</p> <p><b>Flexibility:</b></p>

		<p>The system allows integration of additional models and features without major structural changes.</p> <p><b>Interoperability:</b></p> <p>The system can interact with external software tools and datasets through standard APIs and file formats.</p> <p><b>Maintainability:</b></p> <p>Modular architecture enables easy debugging, updating, and system maintenance.</p> <p><b>Portability:</b></p> <p>The application can run on multiple operating systems supporting Python environments.</p> <p><b>Reliability:</b></p> <p>The system ensures consistent processing results under normal operating conditions.</p> <p><b>Reusability:</b></p> <p>Developed modules such as preprocessing, classification, and clustering can be reused in other NLP applications.</p> <p><b>Robustness:</b></p> <p>The system can handle incomplete or noisy requirement text inputs effectively.</p> <p><b>Testability:</b></p> <p>Individual modules can be tested independently using predefined datasets.</p> <p><b>Usability:</b></p> <p>A simple web interface enables easy interaction for both technical and non-technical users.</p>
5	Other Requirements	<ul style="list-style-type: none"> <li>• The system should support standard text file formats (PDF, TXT, DOCX).</li> <li>• The application should allow exporting processed requirement reports.</li> <li>• The system should support future scalability for handling larger datasets.</li> </ul>
Appendix A: Glossary		<ul style="list-style-type: none"> <li>• <b>NLP:</b> Natural Language Processing</li> <li>• <b>ML:</b> Machine Learning</li> <li>• <b>SRS:</b> Software Requirement Specification</li> <li>• <b>Requirement Classification:</b> Categorizing requirements into functional and non-functional types</li> </ul>

	<ul style="list-style-type: none"><li>• <b>Requirement Clustering:</b> Grouping similar requirements based on similarity</li><li>• <b>Requirement Prioritization:</b> Ranking requirements based on importance or impact</li></ul>
Appendix B: Analysis Model	The project uses an <b>NLP-based requirement processing model</b> consisting of text preprocessing, feature extraction, machine learning classification, clustering, prioritization, and summarization modules forming the complete requirement processing pipeline.
Appendix C: Issues List	<ul style="list-style-type: none"><li>• Dataset availability and quality may affect model accuracy</li><li>• Requirement ambiguity may reduce classification precision</li><li>• Model performance may require periodic retraining with updated datasets</li></ul>