



International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

AI Requirement Traceability System

Ms. S.Evangeline Aishwarya¹, Sanjay R², Santhosh P³, Theepan T K⁴, Varun M⁵, Yashvanth P M⁶

Department of Computer Science and Engineering, Bachelor of Engineering, Sri Shakthi Institute of Engineering and Technology, Coimbatore-641062

ABSTRACT :

The AI Requirement Traceability System is an intelligent web-based application developed to simplify and automate software requirement management throughout the Software Development Life Cycle (SDLC). In conventional software development, requirement traceability is frequently maintained through spreadsheets and separate documents, making it difficult to maintain links between requirements, design artifacts, source code and test cases.

The proposed system applies Artificial Intelligence (AI) and Natural Language Processing (NLP) to analyze Software Requirement Specification documents, extract structured requirements and support automatic traceability mapping. The system provides a Requirement Traceability Matrix (RTM), requirement status tracking and change-impact analysis to identify related files, modules and test cases when a requirement changes. A centralized web dashboard provides project monitoring, AI analysis, traceability and role-based management.

The implemented interface demonstrates requirement upload, AI analysis, project management, traceability monitoring, workflow management and modular system functions. The system is intended to reduce repetitive manual work, improve visibility of requirement coverage and support more consistent software development and testing. Requirement Traceability, Artificial Intelligence, Natural Language Processing, Requirement Traceability Matrix, Impact Analysis, Software Engineering, SDLC, MongoDB.

INTRODUCTION:

Software projects are developed from requirements that describe the expected functions, constraints and behavior of a system. During the Software Development Life Cycle, these requirements are transformed into design artifacts, source code and test cases. Maintaining a clear relationship between the original requirement and these downstream artifacts is important because it helps development teams verify that requirements have been implemented and tested.

Traditional requirement traceability is commonly performed with spreadsheets, documents or manually maintained traceability matrices. Requirement changes also create a need to identify affected modules, source files and test cases. Without a systematic traceability mechanism, a project may contain missing links, incomplete implementation or untested functionality.

The proposed AI Requirement Traceability System addresses these challenges by combining AI and NLP with a web-based traceability environment. The system accepts requirement documents, extracts structured requirements, associates them with software artifacts and provides a traceability matrix. The project interface is organized around dashboards, projects, requirement upload, AI analysis, traceability matrix, impact analysis, reports and user management.

OBJECTIVE:

- To develop an AI-powered system for software requirement management.
- To analyze requirement documents using Artificial Intelligence and Natural Language Processing.
- To extract and organize software requirements automatically.
- To establish relationships between requirements, design documents, source code and test cases.
- To generate and maintain a Requirement Traceability Matrix (RTM).

LITERATURE SURVEY:

1. Ramesh and Jarke studied requirements traceability from an empirical perspective and developed reference models for organizing traceability links across software development activities. Their work established the importance of structured traceability relationships between stakeholder requirements and development outputs.

2. Hayes, Dekhtyar and Osborne investigated requirements tracing as an information-retrieval problem. Their work focused on improving recall and precision so that analysts could discover more useful traceability links while reducing irrelevant links and manual effort.
3. Cleland-Huang, Settimi, Duan and Zou investigated dynamic requirements traceability using supporting evidence. Their work addressed the maintenance difficulty of traditional traceability matrices and explored retrieval strategies to improve trace-link performance.
4. Gotel and Finkelstein analyzed the requirements traceability problem and highlighted the need to follow requirements through their lifecycle in both forward and backward directions. Their work provides a conceptual foundation for linking requirements with downstream development artifacts.
5. Cleland-Huang, Gotel and Zisman and other contributors to software traceability research consolidated traceability practices across the software development lifecycle. The work describes traceability as the ability to relate software artifacts and highlights its role in validation, evolution, reuse and change-impact analysis.

METHODOLOGY:

- 1.Requirement Input and Upload:** Users upload requirement specifications through the web interface. The prototype indicates support for PDF, DOCX and TXT documents and provides a dedicated upload page.
- 2. AI/NLP Requirement Extraction:** The uploaded specification is processed by the AI analysis component. The system segments candidate requirements, classifies category and priority, scores extraction confidence and links requirements to existing artifacts.
- 3. Traceability Mapping:** Extracted requirements are associated with design documents, source-code artifacts and test cases. These relationships are used to create a Requirement Traceability Matrix and provide requirement coverage information.
- 4. Impact Analysis:** When a requirement is modified, the system uses traceability relationships to identify related files, modules, APIs, tables or tests that may require review. The public interface describes this function as an Impact Analysis Graph/Impact Radar.
- 5. Verification and Monitoring:** Requirement status, implementation state and testing information are presented through dashboards and reports. Role-based access is provided for administrators, managers, developers and testers.
- 6. Data Management:** MongoDB is used as the project database. Node.js and Express.js provide backend services, while Python is used for the AI processing layer. The user interface is implemented using HTML, CSS and JavaScript.

EXISTING SYSTEM:

1.Manual Traceability:

Requirements are frequently connected to design, code and tests using spreadsheets or separate documents.

2. Time-Consuming RTM Maintenance:

Updating a traceability matrix requires repeated manual work whenever project artifacts change.

3. Difficult Change Impact Identification:

Developers must manually identify all affected files, modules and test cases after a requirement change.

4. Risk of Missing Links:

Manual traceability can result in incomplete relationships, missing requirements or untested features.

5. Limited Centralized Visibility:

Requirement status, project progress and traceability information may be distributed across multiple documents and tools.

6. Scalability Problem:

The effort required for traceability increases as the number of requirements and development artifacts grows.

PROPOSED SYSTEM:

1.AI Requirement Extraction:

AI/NLP processing extracts structured requirements from uploaded specifications.

2. Requirement Hub:

A centralized requirement area supports requirement ownership, versioning and status management.

3. RTM Engine:

The system generates the Requirement Traceability Matrix and provides matrix export functionality.

4. Impact Radar:

Dependency-oriented impact analysis helps visualize components that may be affected by requirement changes.

5. Verification Suite:

Requirements can be associated with test cases, execution results and defect-related information.

6. Insight Studio:

Interactive dashboards provide project and release-level visibility.

7. Governance Center:

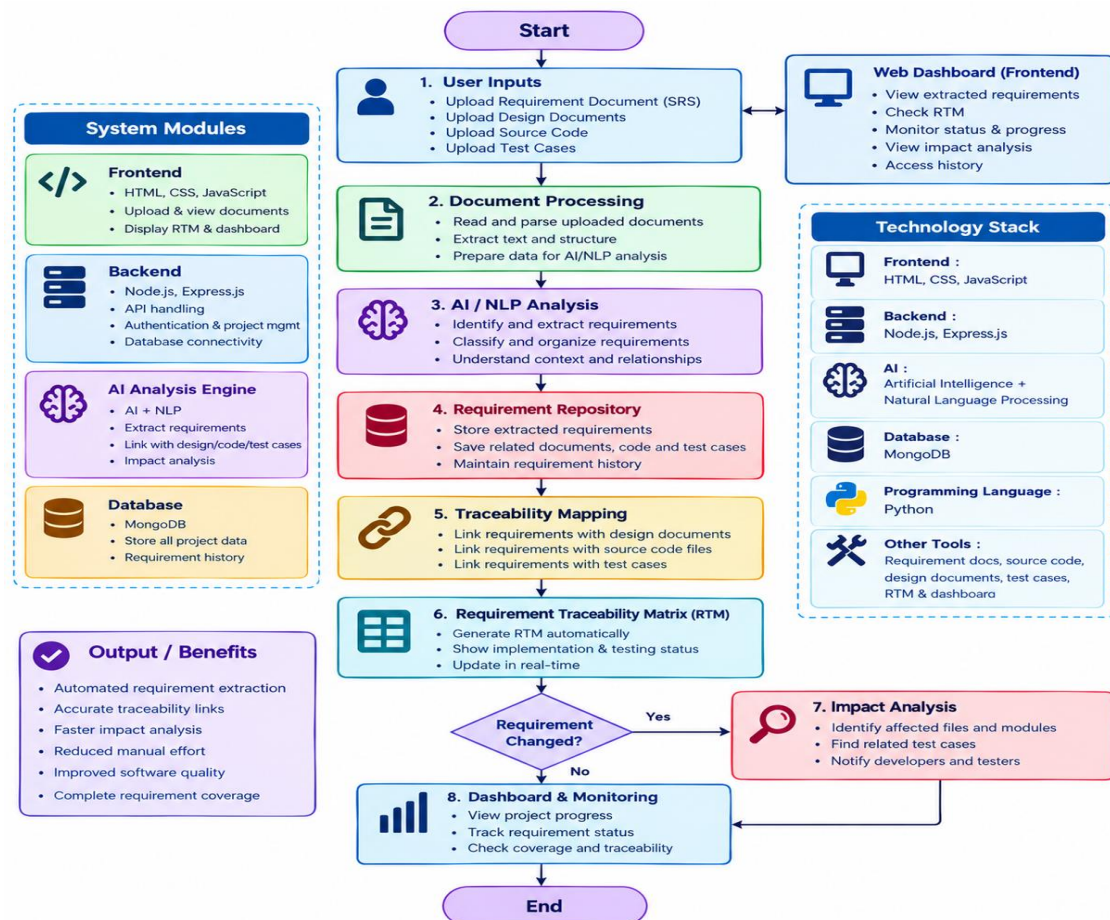
The prototype includes role and permission management, API-key/audit-oriented controls and user administration.

SYSTEM REQUIREMENTS:**Hardware Requirements:**

- Processor: Intel Core i3 or higher or equivalent.
- RAM: Minimum 4 GB; 8 GB recommended.
- Storage: 256 GB or higher.
- Display: Standard HD display or higher.
- Network: Stable internet connection for development and AI/API communication where applicable.

Software Requirements:

- Operating System: Windows 10/11, Linux or macOS.
- Frontend: HTML, CSS and JavaScript.
- Backend: Node.js and Express.js.
- AI Processing: Python with Artificial Intelligence and Natural Language Processing components.
- Database: MongoDB.
- Browser: Google Chrome, Mozilla Firefox or Microsoft Edge.

PROCESSING:

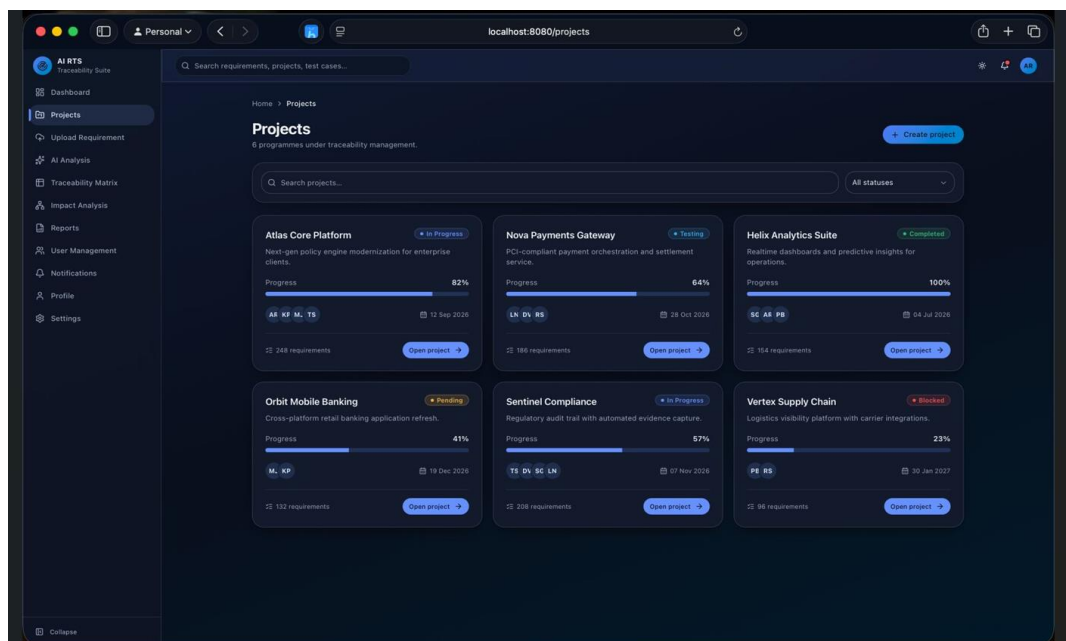
MODULES:

1. *Requirement Management*: Creates, stores, updates, and tracks software requirements.
2. *Document Upload*: Uploads and processes PDF, DOCX, and TXT specification files.
3. *AI Requirement Extraction*: Extracts, classifies, and assigns confidence scores to requirements.
4. *Traceability Matrix (RTM)*: Links requirements with design, source code, and test cases.
5. *Impact Analysis*: Identifies requirement dependencies and possible change impacts.
6. *Verification and Testing*: Links test cases with requirements and tracks testing status.
7. *Dashboard and Reporting*: Displays project progress, requirement status, coverage, and reports.
8. *User Management*: Provides role-based access for Admin, Manager, Developer, and Tester.

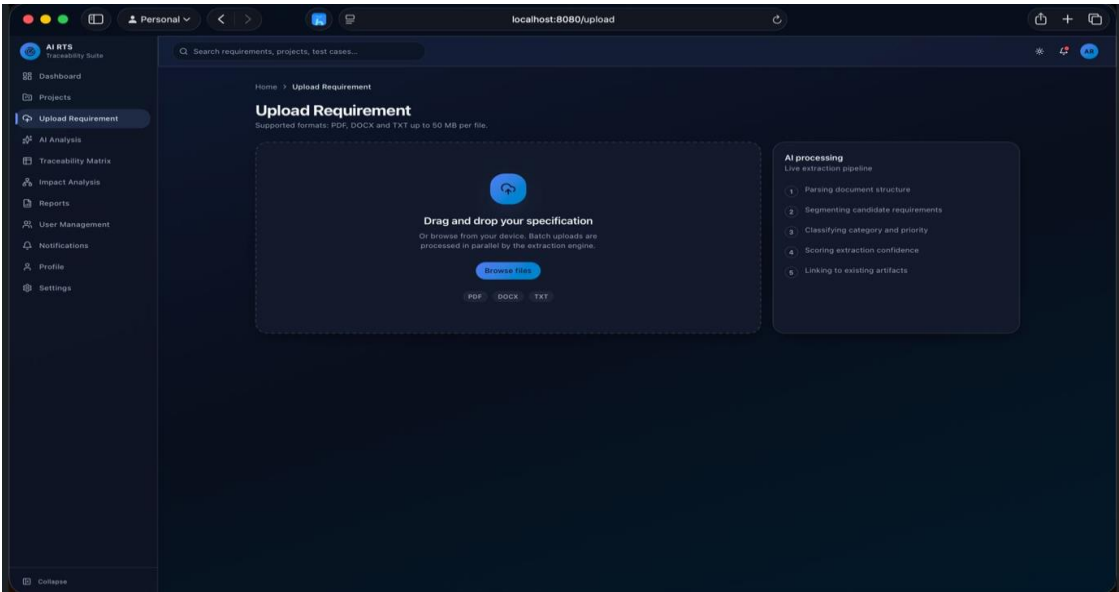
Requirement Traceability Dashboard:



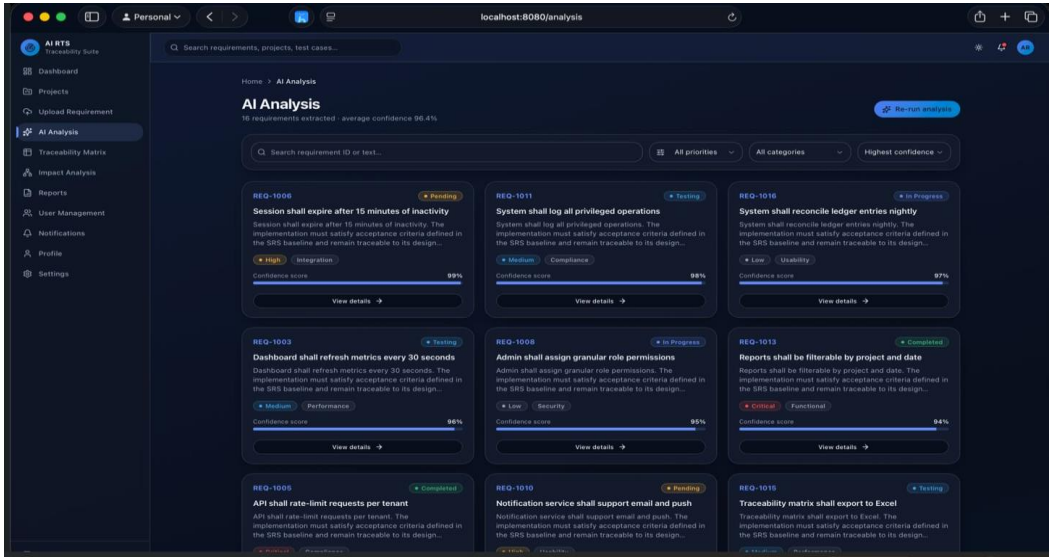
Project and Requirement Management:



Requirement Document Upload and Processing:



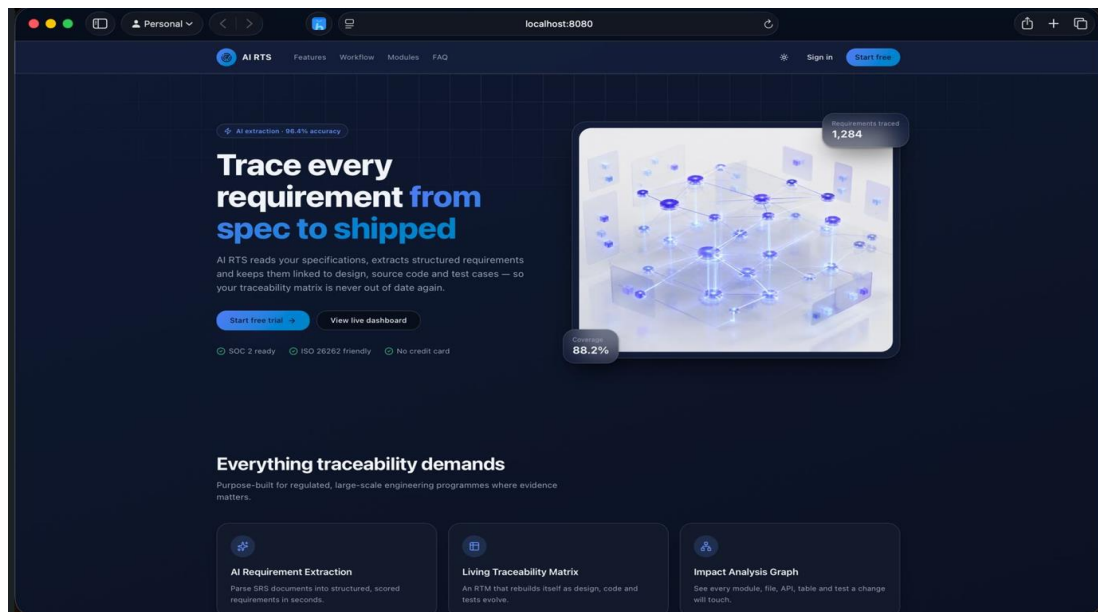
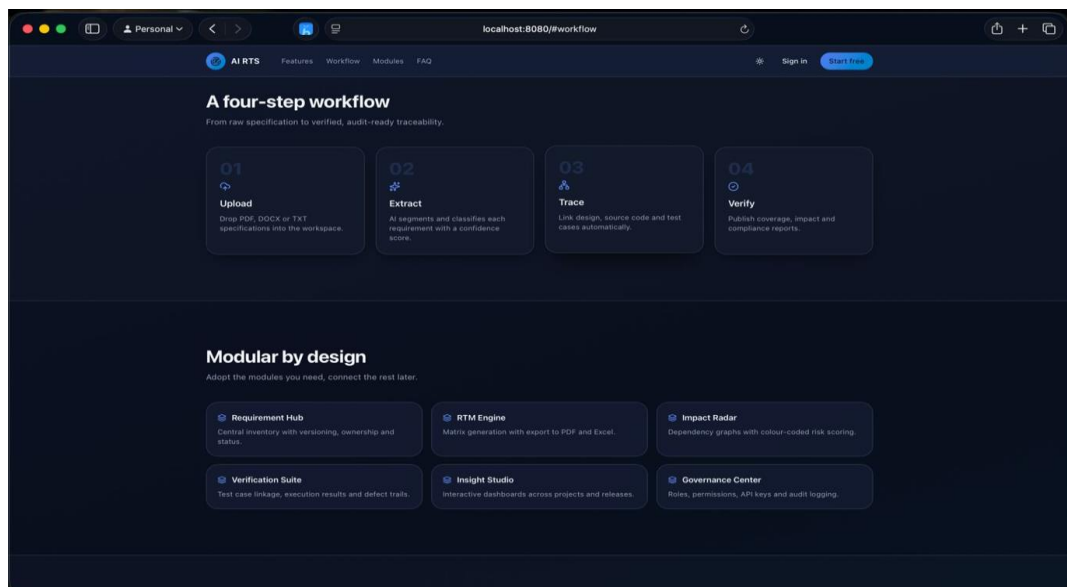
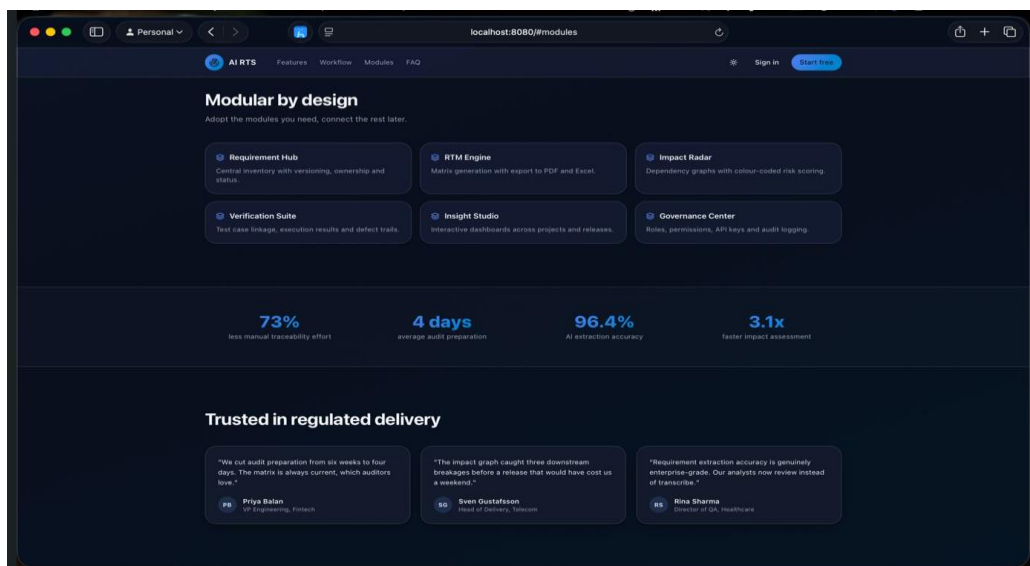
AI-Based Requirement Extraction and Analysis:



User Management and Role-Based Access Control:

The screenshot shows the 'User Management' page. It displays four user roles: Admin (4 users), Manager (9 users), Developer (27 users), and Tester (16 users). Each role has a list of permissions. Below the roles is a 'Permissions matrix' table showing effective capabilities per role. The table has columns for CAPABILITY, ADMIN, MANAGER, DEVELOPER, and TESTER. The capabilities listed are: View dashboard, Create project, Upload requirements, Run AI analysis, Approve RTM, Manage test cases, Manage users & roles, and Export compliance report. The 'Members' section is partially visible at the bottom.

CAPABILITY	ADMIN	MANAGER	DEVELOPER	TESTER
View dashboard	✓	✓	✓	✓
Create project	✓	✓	✗	✗
Upload requirements	✓	✓	✓	✗
Run AI analysis	✓	✓	✓	✗
Approve RTM	✓	✓	✗	✗
Manage test cases	✓	✗	✗	✓
Manage users & roles	✓	✗	✗	✗
Export compliance report	✓	✓	✗	✓

AI Requirement Traceability System – Main Interface:**Requirement Traceability Workflow:****System Modules and Functional Components:**

CONCLUSION:

The AI Requirement Traceability System provides an integrated approach to software requirement management by combining AI/NLP-based analysis with requirement traceability, project monitoring and impact analysis. The system addresses limitations of manual traceability by providing a centralized web interface through which requirements can be uploaded, extracted, linked and monitored.

The prototype demonstrates the major workflow stages of Upload, Extract, Trace and Verify. Its dashboard, AI analysis, project, user-management and modular interfaces provide visibility into requirement status, traceability coverage and project activities. The system uses HTML, CSS and JavaScript for the frontend, Node.js and Express.js for backend services, Python for AI processing and MongoDB for data storage.

The implemented prototype establishes a foundation for automated requirement traceability. Future integration with source-code repositories, testing tools and richer semantic analysis can further strengthen the system and make it suitable for larger software development environments.

REFERENCES:

- [1] B. Ramesh and M. Jarke, "Toward Reference Models for Requirements Traceability," *IEEE Transactions on Software Engineering*, vol. 27, no. 1, pp. 58–93, 2001, doi:10.1109/32.895989.
- [2] I. Sommerville, *Software Engineering*, 10th ed., Pearson, 2015.
- [3] J. Cleland-Huang, O. Gotel, and A. Zisman, Eds., *Software and Systems Traceability*. Springer London, 2012, doi:10.1007/978-1-4471-2239-5.
- [4] J. Cleland-Huang, R. Settimi, C. Duan, and X. Zou, "Utilizing Supporting Evidence to Improve Dynamic Requirements Traceability," *Proceedings of the 13th IEEE International Conference on Requirements Engineering (RE 2005)*, pp. 135–144, 2005, doi:10.1109/RE.2005.78.
- [5] J. H. Hayes, A. Dekhtyar, and J. Osborne, "Improving Requirements Tracing via Information Retrieval," *Proceedings of the 11th IEEE International Requirements Engineering Conference (RE 2003)*, pp. 138–147, 2003, doi:10.1109/ICRE.2003.1232745.
- [6] O. C. Z. Gotel and C. W. Finkelstein, "An Analysis of the Requirements Traceability Problem," *Proceedings of the First International Conference on Requirements Engineering*, pp. 94–101, 1994.