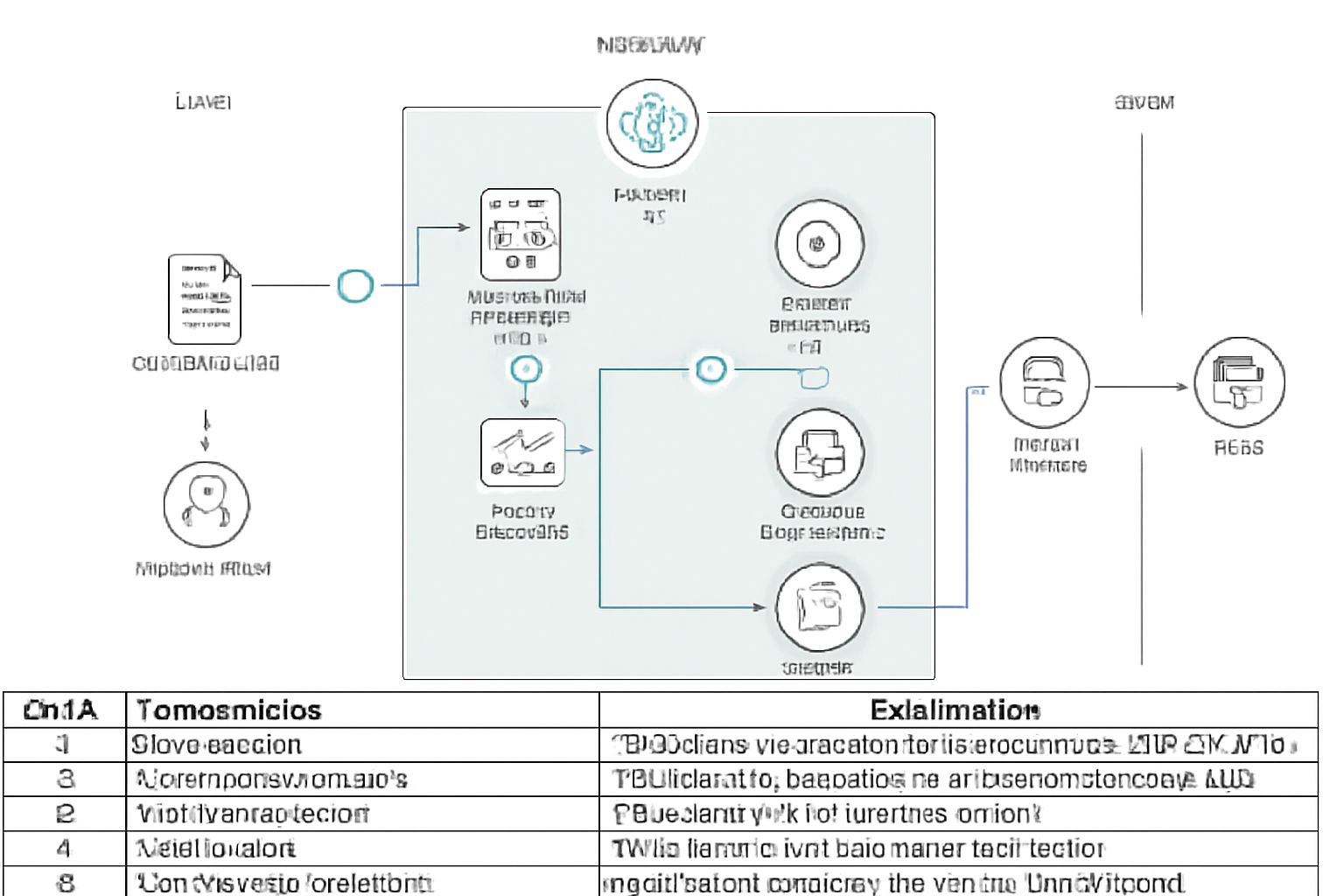
**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 31 January 3035 |
| Team ID | LTVIP2025TMID32689 |
| Project Name | smartSDLC-AI-enhanced software develpoment life cycle |
| Maximum Marks | 4 Marks |

# Technical Architecture

The architecture for SmartSDLC leverages AI and NLP to automate key tasks in the software development lifecycle. The platform integrates with third-party APIs, cloud infrastructure, and machine learning models to automate the process from requirement gathering to code generation and bug fixing. Key components include a web UI for user interaction, cloud storage for data management, and AI models for intelligent code generation, bug fixing, and real-time support.



# Table-1: Components & Technologies

|  |  |  |  |
| --- | --- | --- | --- |
| S.No | Component | Description | Technology |
| 1 | User Interface | Web UI for user interaction | HTML, CSS, JavaScript / React.js |
| 2 | Application Logic-1 | NLP-based requirement analysis logic | Python, IBM Watson NLP |
| 3 | Application Logic-2 | Code generation logic based on user stories | Python, Custom AI Algorithms |
| 4 | Application Logic-3 | Bug fixing logic using AI models | Python, IBM Watson AI |
| 5 | Database | Structured data storage for stories, code, and test cases | MySQL / NoSQL |
| 6 | Cloud Database | Cloud-hosted database | IBM Cloudant / AWS DynamoDB |
| 7 | File Storage | File storage for assets | IBM Block Storage, Cloud Storage |
| 8 | External API-1 | External API for authentication | OAuth API (Gmail, LinkedIn) |
| 9 | External API-2 | API for project management and task integration | Jira API |
| 10 | Machine Learning | AI models for code generation and bug fixing | IBM Watson AI, Custom ML Models |
| 11 | Infrastructure | Cloud infrastructure for hosting SmartSDLC | IBM Cloud, AWS, Kubernetes |

# Table-2: Application Characteristics

|  |  |  |  |
| --- | --- | --- | --- |
| S.No | Characteristics | Description | Technology |
| 1 | Open-Source Frameworks | Frameworks used for development | React.js, Node.js, Flask, TensorFlow |
| 2 | Security Implementations | Security features to protect user data and privacy | SHA-256, OAuth 2.0, SSL/TLS, IAM Controls |
| 3 | Scalable Architecture | Scalability justification (microservices, 3-tier architecture) | Kubernetes, Cloud Foundry, Docker |
| 4 | Availability | Ensures the system is available with minimal downtime | Load Balancers, Distributed Servers |
| 5 | Performance | Performance considerations for handling large workloads | Cache, CDN, Horizontal Scaling, Load Balancers |