# Project Design Phase-II Technology Stack (Architecture & Stack)

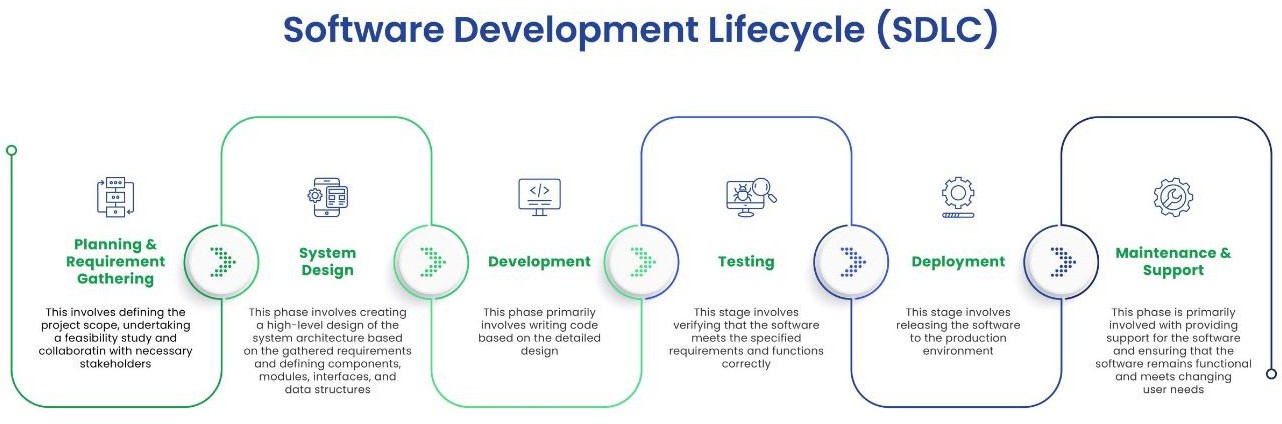
|  |  |
| --- | --- |
| Date | 26 June 2025 |
| Team ID | LTVIP2025TMID60019 |
| Project Name | SmartSDLC – AI-Enhanced Software  Development Lifecycle |
| Maximum Marks | 4 Marks |

**Technical Architecture:**

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

# Example: AI-Augmented SmartSDLC for Agile Software Teams

**Reference:** https://aws.amazon.com/blogs/apn/transforming-the-software-development- lifecycle-sdlc-with-generative-ai/



**Table-1 : Components & Technologies:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1. | User Interface | Web dashboard for managing  projects and viewing AI insights | React.js, HTML5, CSS3 |
| 2. | Application Logic-  1 | AI-based requirement analysis  and document parsing | Python (Flask), SpaCy,  NLTK |
| 3. | Application Logic-  2 | Transcribe voice meetings for  task logging | IBM Watson Speech to  Text |
| 4. | Application Logic-  3 | Chatbot support for answering  SDLC queries | IBM Watson Assistant |
| 5. | Database | Stores user data, project | MongoDB (NoSQL), |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | metadata, task logs | MySQL |
| 6. | Cloud Database | Cloud-hosted database for real-  time syncing | IBM Cloudant |
| 7. | File Storage | Code files, generated reports,  documentation | IBM Block Storage,  Local Filesystem |
| 8. | External API-1 | GitHub integration for CI/CD &  code analysis | GitHub REST API |
| 9. | External API-2 | Integration with Jira for agile  boards and ticketing | Jira API |
| 10. | Machine Learning Model | Predict bugs, generate code suggestions, and estimate effort | TensorFlow, OpenAI Codex, Scikit-learn |
| 11. | Infrastructure (Server / Cloud) | Cloud-native deployment with CI/CD pipeline | Kubernetes, Docker, IBM Cloud Foundry,  Jenkins |

**Table-2: Application Characteristics:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 1. | Open-Source  Frameworks | Frameworks used for backend,  frontend, and ML | Flask, React.js,  TensorFlow, Kubernetes |
| 2. | Security Implementations | Role-based access, data encryption, secure APIs | JWT, SHA-256, OAuth2.0, HTTPS, IAM  Policies |
| 3. | Scalable  Architecture | Microservice-based deployment  for each SDLC phase | Kubernetes, Docker |
| 4. | Availability | Ensured with replicated services and cloud load  balancer | NGINX, IBM Cloud Load Balancer, Multi-Zone  Setup |
| 5. | Performance | Use of Redis for caching,  Celery for background tasks, CDN for static files | Redis, Celery, Cloudflare CDN |

**References:**

https://developer.ibm.com/patterns/ai-powered-devops/

https://[www.ibm.com/cloud/cloudant](http://www.ibm.com/cloud/cloudant)

https://[www.ibm.com/cloud/watson-speech-to-text](http://www.ibm.com/cloud/watson-speech-to-text)

https://docs.github.com/en/rest

https://developer.atlassian.com/cloud/jira/platform/rest/v3/

https://c4model.com/

https://[www.ibm.com/cloud/architecture](http://www.ibm.com/cloud/architecture)