**Development Phases & Team Management**

To ensure timely delivery and full coverage of all core and premium features, the project follows an Agile Scrum methodology broken into **5 sprint phases**. Each phase consists of specific technical deliverables, review checkpoints, and team collaboration milestones.

**📅 Phase 1: Planning & Setup (Week 1–2)**

**Goals:**

* Establish architecture and infrastructure.
* Define tech stack and core libraries.
* Assign roles and responsibilities.

**Deliverables:**

* Project charter and task board (Trello/ClickUp).
* Repository setup (GitHub/GitLab) with branching strategy.
* Database schema draft (PostgreSQL + MongoDB/Redis integration plan).
* FastAPI scaffolded project structure.
* Requirements documentation refined from NSED742 brief.

**Team Roles:**

* Lead Developer: Design system architecture.
* DevOps Lead: Setup CI/CD pipelines.
* Backend Developer: Schema proposal & ORM selection.
* Product Owner: Refine use cases with instructor feedback.

**Milestones:**

* Architecture review checkpoint.
* NFRs (non-functional requirements) defined.

**🛠️ Phase 2: Backend Core APIs (Week 3–5)**

**Goals:**

* Implement user auth, tender upload, profile setup, and team collaboration APIs.

**Core Features:**

* JWT-based Authentication & Role-Based Access Control (Admin, SME, Collaborator)
* Tender CRUD operations (metadata, documents)
* Team workspace setup (project links, shared notes)
* Profile setup & scoring inputs (e.g., CIDB, BEE level)
* Public API endpoint for listing tenders

**Testing:**

* Swagger/OpenAPI documentation for endpoints
* Pytest unit tests for endpoint logic

**Additions:**

* Implement Redis caching for public tenders API.
* Use PostgreSQL JSONB for semi-structured fields.

**Milestones:**

* Backend milestone review
* API integration tests

**🤖 Phase 3: AI Features & Analytics (Week 6–8)**

**Goals:**

* Integrate AI summarization, readiness scoring, and dashboard analytics.

**AI Features:**

* Transformer-based summarizer (T5/BART)
* Rule-based readiness scoring algorithm
* Keyword matching + fuzzy logic for compliance indicators

**Analytics:**

* Buyer spending insights (by buyer, category, date)
* SME performance tracking (tender success rate, matched sectors)

**Data Sources:**

* MongoDB for AI-generated metadata
* PostgreSQL for structured analytics data

**Additions:**

* Hybrid scoring pipeline combining keyword scores + eligibility heuristics
* Async job queue for large summarization tasks (e.g., Celery + Redis)

**Milestones:**

* AI model evaluation review
* Dashboard prototype completed

**🎨 Phase 4: Frontend UI + Integration (Week 9–10)**

**Goals:**

* Build modern responsive UI and integrate with backend endpoints.

**Stack:**

* React or Vue.js with Tailwind CSS
* Axios + JWT token storage

**UI Modules:**

* Login/Registration (with user role routing)
* Tender board + filtering by sector, value, deadline
* Team dashboard (shared workspace)
* Tender details + summarization + scoring display
* Export Report UI (Pro feature prompt for free users)

**Additions:**

* WebSocket integration for live notifications (e.g., team notes, deadlines)

**Milestones:**

* Usability testing with pilot users
* Integration demo session with instructor

**🧪 Phase 5: Testing, Security, Export (Week 11–12)**

**Goals:**

* Finalize and secure the application, perform full testing and export features.

**Testing:**

* Unit, integration, and end-to-end (Playwright/Cypress)
* Load testing for public API
* AI summarization failure case testing

**Security Practices:**

* OWASP Top 10 checklist
* JWT expiration and refresh handling
* MongoDB/PostgreSQL injection prevention

**Export Report (Pro Tier):**

* PDF/CSV export engine
* Conditional logic: Export allowed for Pro users only
* Include: Scoring breakdown, summary, deadlines, buyer insights

**Additions:**

* Logging via FastAPI middleware
* Rate-limiting for public API routes
* Admin interface for managing flagged tenders

**Milestones:**

* Final deployment to cloud (Render/Heroku/Docker on GCP)
* Instructor final evaluation