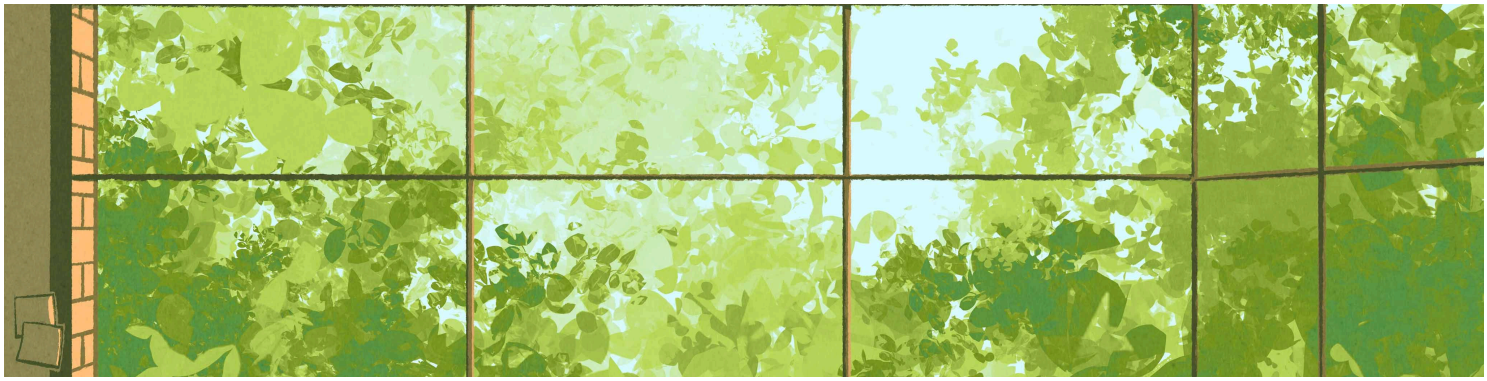


Edbot Product PRD

📘 Product Requirements Document (PRD)

Product Name: Dikkha AI (not final)

Tagline: *"পড়াশোনা এখন বোঝার মজা" - (need to change, AI generated)*
Version: Beta (Class 9–10 only)
Date: November 2025
Owner: Risad Malik, Founder & AI Engineer
Developers: Nafis (full stack) , Mrittika (AI)

---

1. 🎯 Product Overview

1.1 Mission

To make Bangladeshi students truly *understand* their textbooks instead of memorizing them — by turning every NCTB line into an **interactive learning experience** powered by AI.

1.2 Problem Statement

Bangladeshi Class 9–10 students rely heavily on rote memorization, coaching centers, and guidebooks due to:

- Lack of conceptual explanations in the NCTB books
- Limited access to quality tuition.
- Static learning materials (text-only, non-interactive).

1.3 Solution

An AI-powered mobile application that:

- Displays digital NCTB books in a reader interface.
- Allows students to select any text and instantly:
  - Get an LLM-based explanation in Bengali.
  - Generate quizzes and flashcards.

    - Listen to an audio/podcast version.
    - Take notes and track their learning progress.
- Uses gamification and scoring to increase motivation.

---

2. 📦 Scope (Beta)

In Scope (Beta):

- Class 9–10 NCTB books (one subject initially, e.g., Physics).
- Interactive Reader with AI Explain, Quiz, Flashcard, Note, and Podcastify.
- User authentication and personalized dashboard.
- Gamified scoring and XP system.
- Learning progress tracking and analytics.
- Offline reading (text only).
- AI Chat (limited RAG-based Q&A).

Out of Scope (For Future Phases):

- Teacher or guardian dashboard.
- Peer study groups / leaderboard.
- AR/interactive diagrams.
- Marketplace for teacher-created content.

---

3. 👥 Target Users

User Type	Description	Key Needs
Students (Primary)	Class 9–10, mobile-first learners	Conceptual clarity, quick help, motivation
Parents (Secondary)	Monitor progress (future)	Affordable support tool
Schools / Tuition Centers (Partners)	Validate tool efficacy	Engagement and test improvement data

4. 🔄 User Journey

1. **User installs app → Login/Sign-up**
    - Google / Phone auth (Firebase)
1. **Selects Class → Subjects → Board**
2. **Home Dashboard**
    - See progress, streaks, quick links to books
1. **Opens Book → Reads Text**
2. **Highlights text → chooses:**
    - **Explain:** AI explanation
    - **Quiz Me:** Generate MCQs
    - **Flashcard:** Create summary Q&A
    - **Podcastify:** Listen to audio
    - **Note:** Save side notes
1. **Completes session → earns XP**
2. **Reviews Flashcards/Notes later**
3. **Checks Progress Dashboard**
4. *(Optional)* Uses “Ask Shikkhok” chat for open-ended queries.

---

5. 🧩 Core Features (Detailed Requirements)

5.1 Authentication

- Firebase Auth integration (Google & phone-based OTP).
- Minimal onboarding (choose class, board, subjects).

Store user profile (JSON):

```
{
 "user_id": "uuid",
 "class": "9",
 "board": "Dhaka",
 "subjects": ["Physics", "Biology"],
 "xp": 230,
 "streak_days": 4
}
```

-

5.2 Book Library

Purpose: Organized access to NCTB books.

Requirements:

- Books displayed as subject cards.
- Each subject opens chapter list.
- Chapter cards show title + progress bar.

Data Source:

- Locally hosted EPUB/PDF files.
- Metadata JSON for chapter mapping.

5.3 Interactive Reader (Core)

Reader Layout

Section	Function
Main Text	Scrollable NCTB chapter text
Selection Tools	Floating menu: Explain
Side Drawer (optional)	Notes, Flashcards, or Quiz view
Embedded Images	Fetched via OCR or preloaded assets

Reader Actions

(a) Explain

Highlighted text → send to backend API:

`POST /explain`

`body: { text, subject, class, chapter_id }`

Response:

```
{
 "simplified_explanation_bn": "...",
 "example": "...",
```

```
 "confidence": 0.92,
 "source_reference": "Chapter 1, Para 3"
}
```

- UI shows Bengali text + source + “Did you understand?” (Yes/No).

(b) Quiz Me

- Generates 3–5 MCQs per selection.
- Store in quiz DB with difficulty tag.
- After submission:
  - Show correct answers + explanation.
  - Update XP score.
  - Option to “Add to Flashcards.”

(c) Flashcard

- Creates Q&A pair from selected text.

Stored as JSON:

```
{ "question": "What is reflection?", "answer": "..." }
```

- Available in Flashcard Hub for review.

(d) Note

- Opens side panel → user types.
- Notes auto-saved under chapter.

(e) Podcastify

- Converts selection or explanation text → Bengali TTS audio.
- Uses Sonix / PlayHT API.
- Stored with title, subject, and playback duration.

5.4 Flashcards Hub

- View all flashcards organized by subject → chapter.
- Daily review challenges (e.g., “Review 10 cards”).
- Flip animation (front/back).
- Progress: “Reviewed 60% of cards this week.”

5.5 Quiz Mode

- Choose chapter → take quiz.
- Timer mode (optional).
- XP system updates on score.

- Display analytics (accuracy per topic).

5.6 Notes Section

- Group by subject → chapter.
- Each note editable + deletable.
- Option: “Summarize notes” (LLM summary endpoint).
- Export as PDF (Phase 2).

5.7 Progress Dashboard

Metrics displayed:

- Chapters completed (%)
- Avg quiz accuracy
- Flashcards reviewed
- Time spent (mins)
- Current streak 🔥
- Total XP and level

Visualizations:

- Line graph for daily XP
- Bar chart per subject progress
- Badges grid (e.g., “Quiz Master”, “Consistent Learner”)

5.8 Gamification System

Trigger	XP	Description
Correct quiz answer	+5	Adds XP
3-answer streak	+2	Bonus XP
Reading full chapter	+10	—
Creating flashcard	+3	—
Daily login	+5	—
Finishing goal	+20	Unlock badge

Levels:

- 0–100 XP → Beginner
- 100–300 → Explorer
- 300–600 → Master
- 600+ → Shikkhok Pro

Data Model:

```
{
 "xp": 230,
 "level": "Explorer",
 "badges": ["Curious Mind", "Quiz Master"],
 "streak_days": 4
}
```

5.9 Ask Shikkhok (AI Chat Mode)

Function: Free-form RAG chat for user questions.

Process:

1. User enters question.
2. System retrieves relevant textbook chunks (Weaviate vector search).
3. Sends top 3 passages + question to Gemini 2.5 Flash.
4. Returns Bengali answer + source citations.
5. Option: “Save as Note” or “Generate Quiz.”

---

6. ⚙️ Technical Architecture

Layer	Tool / Framework
Frontend	Flutter (cross-platform)
Backend	Node.js (NestJS / Express)
Database	Firebase Firestore / Supabase
AI / LLM	Gemini 2.5 Flash
RAG Vector Store	Weaviate
OCR	Mathpix

TTS / Audio	Sonix or PlayHT (Bengali voices)
Auth	Firebase Authentication
Storage	Google Cloud / Supabase Storage
Analytics	Firebase / Mixpanel

---

7. 📊 Data Model Summary

Entity	Fields
User	id, name, class, xp, level, streak_days
Book	id, subject, chapters[], epub_url
Chapter	id, title, progress, last_read_pos
Flashcard	id, question, answer, subject, chapter_id, reviewed
Quiz	id, question, options[], correct_option, accuracy, timestamp
Note	id, content, chapter_id, created_at
Audio	id, file_url, title, duration

---

8. 🧠 AI Workflow (Backend)

Explain Flow

`User selects text → POST /explain`

`→ Retrieve related chunks from Weaviate`

`→ Prompt Gemini with context`

`→ Return simplified explanation + source`

Quiz Flow

`User clicks “Quiz Me”`

`→ Prompt Gemini with chapter excerpt`

`→ Return JSON array of MCQs`

`→ Save to user DB`

Flashcard Flow

`Selection → Prompt Gemini ("Create Q&A")`

`→ Save locally`

---

9. 🎨 UI / UX Guidelines

Element	Specification
Color palette	Olive green (#90A955), parchment cream (#FAF3DD), accent teal (#4C956C)
Font	Noto Sans Bengali
Buttons	Rounded corners (radius 16–24px)
Animations	Fade/slide transitions, micro feedback
Reader mode	Serif typeface for body text, minimal distractions
Dashboard	Card-based layout, soft shadows, 2-column grid
Icons	Lucide / Material icons (outlined, rounded)

---

10. 📈 Success Metrics (for Beta)

Metric	Target
Daily Active Users (DAU)	200+
Average session duration	15+ minutes

Avg quiz accuracy	≥70%
Daily streak retention	30% of users (7-day)
LLM cost per user	<$0.05/session
Pre/post improvement (pilot test)	+20% in quiz scores

11. 🧩 Non-Functional Requirements

Type	Specification
Performance	95% responses < 2 sec (cached)
Uptime	99% during beta
Offline mode	Cache chapters locally
Security	Firebase Auth, HTTPS only
Privacy	Minimal PII stored
Scalability	Designed for 50K DAU
Accessibility	Supports Bengali TTS + large fonts

12. 🚀 MVP Development Plan (Phases)

Phase	Duration	Deliverables
Phase 1 (Month 1–2)	Setup infra + UI/UX	Login, Book Library, Reader core
Phase 2 (Month 3–4)	AI integration	Explain + Quiz + Flashcard + Notes
Phase 3 (Month 5)	Gamification + Analytics	XP, Badges, Dashboard

Phase 4 (Month 6)	Beta Launch + Pilot Test	Feedback loop, iteration

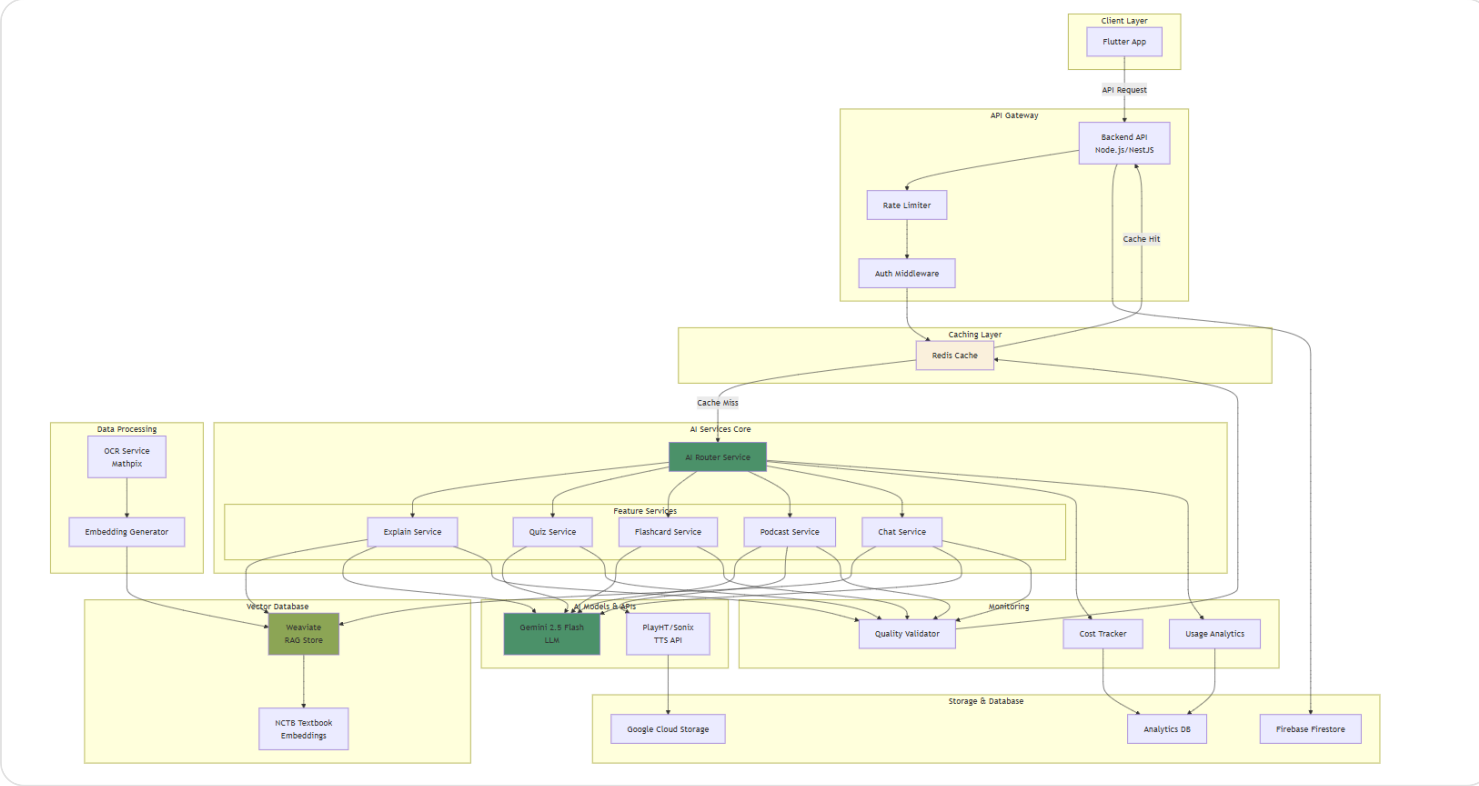
13. ⚠️ Risks & Mitigation

Risk	Mitigation
LLM hallucination	RAG + source citation
High API costs	Cache + batch calls
OCR errors	Manual correction + fallback
Student confusion	Use simple Bengali tone
Low adoption	Pilot with 2 schools + tuition centers
Legal (NCTB content)	Seek licensing / use excerpts under fair use

14. 📋 Future Expansion (Post-Beta)

- Teacher dashboard for assignments.
- Parent progress view.
- Peer leaderboard and community.
- Offline full-mode (text + audio).
- Gamified “Exam Challenges.”
- AI-generated video explainers.
- Adaptive curriculum recommendation.

System Architecture(Need to Finalize)



System flow:

User Journey → System Components:

User Selects Text in Reader
↓
Flutter App sends request
↓
Backend API receives request
↓
Routes to appropriate service:
├— Explain → (Gemini + RAG)
├— Quiz → (Gemini)
├— Flashcard → (Gemini)
├— Podcast → (PlayHT)
└— Chat → RAG (Weaviate) with multimodal input
↓
Response sent back to app
↓
UI displays result
↓
User feedback collected
↓

Analytics tracked

Feature Flow:

User selects text

→ App sends to /api/ai/explain
→ Check cache (Redis)
→ If cached: return immediately
→ If not cached:
    → RAG retrieves context (Weaviate)
    → AI generates explanation (Gemini)
    → Validate quality
    → Cache result
→ Return to user
→ Track cost & usage
→ Store user feedback

Quiz Generation Flow:

User clicks "Quiz Me"

→ App sends text to /api/ai/quiz/generate
→ Check cache
→ If not cached:
→ AI generates MCQs (Gemini)
→ Validate JSON structure
→ Check Bengali quality
→ Cache result
→ Save to user's quiz history
→ Return to app
→ User takes quiz
→ Update XP & analytics

Podcast Generation Flow:

User clicks "Podcastify"

→ App sends text to /api/ai/podcast/generate
→ AI explains text (Gemini)
→ Format as podcast script(Dialog conversion)
→ Generate audio (PlayHT)

→ Upload to storage
→ Return audio URL
→ Track cost (expensive)

Chat Flow (RAG-based):

User asks question in chat
→ App sends to /api/ai/chat
→ RAG searches textbook (Weaviate)
→ Retrieve top 3-5 relevant chunks
→ AI generates answer (Gemini + context)
→ Return answer with sources
→ Save conversation history
→ Track usage

Responsibilities of AI Team:

Components	Assignee	Responsibilities
RAG	AI Team	Weaviate setup, Chunking, Vector search
OCR	AI Team	Mathpix API, Image processing
LLM Integration	AI Team	Gemini API, Prompt engineering, Response parsing (chat, quiz, flashcards)
TTS/STT	AI Team	PlayHT(need testing), Soniox, Audio generation