==========

BONDHON - AI-POWERED STUDENT LIFE OPERATING SYSTEM
Complete Project Plan & Implementation Guide

PROJECT OVERVIEW

Name: Bondhon (Bengali for "companion")
Tagline: Your AI-powered study companion for Bangladeshi students
Target: University & college students (15-25 years) in Bangladesh
Core Value: All-in-one platform for studying, career prep, and wellbeing

WHY THIS IS DIFFERENT FROM COMPETITORS

Maya AI / Other Mental Health Apps → Only focus on emotional support
Bondhon → 10+ unique features covering study, career, finance, community

UNIQUE FEATURES NO ONE ELSE HAS:

1. AI Study Group Matcher
2. Smart Timetable Optimizer
3. Crowd-Sourced Question Bank
4. Document Scanner + AI Note Organizer
5. Skill Gap Analyzer + Learning Paths
6. Mock Interview AI
7. Anonymous Campus Feed
8. Peer Tutoring Marketplace
9. Financial Literacy Dashboard
10. AI Study Companion (enhanced)

==========

PART 1: CORE FEATURES BREAKDOWN

FEATURE 1: AI STUDY GROUP MATCHER

Problem: Students can't find study partners with similar goals/schedules
Solution:

- User creates profile: subjects, study times, learning style, goals
- AI matches students using similarity algorithm
- Suggests optimal meeting times for groups
- Creates group chat with AI moderator
- Tracks group study sessions

Technical Implementation:

- Backend: Python FastAPI

- Backend: Python FastAPI
- Algorithm: Cosine similarity on user vectors
- Real-time chat: Socket.io
- Database: PostgreSQL for user profiles
- Free AI: Ollama Llama 3.1 for group suggestions

User Flow:

1. Student fills profile (10 questions)
2. AI suggests 3-5 potential study partners
3. Student sends join request
4. Group formed → dedicated chat room
5. AI suggests study topics and schedules

Business Value:

- Network effects (more users = better matches)
- High engagement (students return for groups)
- Viral growth (students invite friends)

FEATURE 2: SMART TIMETABLE + HABIT OPTIMIZER

Problem: Students have chaotic schedules and poor time management
Solution:

- Import class schedule (Google Calendar / manual)
- AI analyzes productive hours (when you complete tasks)
- Auto-schedules study blocks in free time
- Pomodoro timer with AI-recommended breaks
- Predicts exam stress periods
- Daily/weekly productivity reports

Technical Implementation:

- Google Calendar API (free) for import/sync
- Python algorithm: constraint satisfaction problem
- Data tracking: time spent per subject
- Visualization: Chart.js for productivity graphs
- AI: Ollama analyzes patterns

Features:

- Smart suggestions: "You're most productive 8-11 PM, schedule hard topics then"
- Adaptive breaks: "You focused 90 min, take 15 min break"

- Exam countdown: "Physics exam in 14 days, need 20 hours prep"
- Habit tracking: Study streaks, completion rates

Business Value:

- Daily usage (checking schedule)
- Clear value proposition (better grades)
- Premium feature (advanced analytics)

FEATURE 3: CROWD-SOURCED QUESTION BANK + AI TUTOR

Problem: Past exam papers hard to find, explanations unclear
Solution:

- Students upload questions (text/image)
- AI categorizes by subject, topic, difficulty
- Search: "Show me calculus integration questions"
- AI generates explanations using free LLM
- Other students can improve answers (Stack Overflow style)
- Gamification: points for contributing

Technical Implementation:

- OCR: Tesseract for image-to-text
- Vector DB: Chroma for semantic search
- AI: Ollama Llama 3.1 for explanations
- Voting system: PostgreSQL
- Category auto-detection: spaCy NLP

User Flow:

1. Student uploads question image
2. OCR extracts text
3. AI categorizes: "Calculus > Integration > Substitution Method"
4. AI checks if similar question exists
5. If new → AI generates explanation
6. Other students can upvote/edit
7. Best answers rise to top

Gamification:

- +10 points for uploading question
- +50 points if your answer gets 10+ upvotes
- Badges: "Calculus Expert" (100 questions answered)

- Leaderboard: Top contributors per subject

Business Value:

- Content generation from users (free)
- High retention (students return to search)
- Premium: unlimited AI explanations

FEATURE 4: DOCUMENT SCANNER + AI NOTE ORGANIZER

Problem: Handwritten notes messy, hard to search, lost easily
Solution:

- Scan notes with phone camera
- OCR converts to searchable text
- AI auto-organizes by subject
- Generates flashcards automatically
- Creates summary sheets before exams
- Share notes with classmates

Technical Implementation:

- Frontend: Camera API for photo capture
- OCR: Tesseract with OpenCV preprocessing
- AI categorization: Ollama analyzes content
- Storage: Supabase storage (free tier)
- Flashcard generation: spaced repetition algorithm

Features:

- Batch scanning: 10 pages at once
- Auto-detect subject from content
- Search within handwritten notes
- Export to PDF
- Collaborative notebooks (share with group)

User Flow:

1. Take photo of notebook page
2. App crops, enhances, OCRs
3. AI: "This looks like Physics - Newton's Laws"
4. Saves to "Physics" folder
5. Extracts key concepts: "F=ma", "Inertia"
6. Creates flashcards automatically

Business Value:

- Daily usage (after every class)
- Free tier: 5 scans/day
- Premium: unlimited scans
- University deals (all students get premium)

FEATURE 5: SKILL GAP ANALYZER + LEARNING PATH

Problem: Students don't know what skills needed for desired jobs
Solution:

- Student selects dream job (Software Engineer, Banker, etc.)
- AI scrapes real job postings from Bdjobs, LinkedIn
- Analyzes required skills
- Student takes quick assessment (20 questions)
- AI shows gap: "You need: Python, SQL, Git"
- Creates personalized learning roadmap
- Curates FREE resources (YouTube, freeCodeCamp, etc.)
- Tracks progress with certificates

Technical Implementation:

- Web scraping: BeautifulSoup + Selenium
- Job boards: Bdjobs.com, LinkedIn, Prothom Alo Jobs
- NLP: Extract skills from job descriptions
- Assessment: Multiple choice + coding challenges
- Course aggregator: API from YouTube, Coursera, edX
- Progress tracking: PostgreSQL

User Flow:

1. "I want to be a Software Engineer"
2. AI: "Analyzing 200 job postings..."
3. Shows: Required Skills (Python 85%, SQL 70%, Git 60%)
4. Assessment: "Take 20-min skills test"
5. Result: "You have Python ✓, need SQL ✗, Git ✗"
6. Learning path: Week 1-2: SQL basics (free course links) Week 3-4: Git fundamentals Week 5-6: Build portfolio project
7. Check off completed courses
8. Retake assessment to see improvement

Features:

- 50+ career paths (engineering, business, medicine, etc.)
- Bangladesh job market data
- Skill trends over time
- Salary expectations by skill level
- Company-specific prep (Grameenphone, BRAC, etc.)

Business Value:

- Career-focused (high perceived value)
- Premium: advanced career insights
- B2B: University career services pay for access
- Affiliate revenue: course sales commissions

FEATURE 6: MOCK INTERVIEW AI

Problem: Students unprepared for job interviews
Solution:

- Choose company/role (Grameenphone, BRAC Bank, etc.)
- AI conducts realistic mock interview
- Text or voice input
- Common questions for that specific role
- AI gives detailed feedback
- Unlimited practice

Technical Implementation:

- LLM: Ollama Llama 3.1 as interviewer
- Voice: Web Speech API (browser-based, free)
- Question bank: Scraped from Glassdoor, company sites
- Feedback algorithm: Analyzes answer quality
- Recording: Optional for self-review

Features:

- Role-specific questions:
  - Software Engineer: "Explain OOP concepts"
  - Marketing: "How would you launch a product?"
  - Banking: "Why investment banking?"
- Behavioral questions: STAR method feedback
- Technical questions: Coding challenges
- Follow-up questions based on answers

- Performance score: 1-10 with improvement tips

User Flow:

1. Select: "Mock Interview for Software Engineer at bKash"
2. AI: "Hi, I'm your interviewer. Tell me about yourself."
3. Student answers (text/voice)
4. AI asks 8-10 questions (30 min session)
5. End: AI gives detailed feedback
    - Communication: 7/10 (too many filler words)
    - Technical knowledge: 8/10 (solid answers)
    - Confidence: 6/10 (improve body language)
6. Suggestions: "Practice the STAR method"

Business Value:

- High-value premium feature
- Career prep is paid market
- Partner with recruitment agencies

FEATURE 7: ANONYMOUS CAMPUS FEED

Problem: Students can't discuss sensitive issues publicly
Solution:

- Post anonymously about campus topics
- AI moderates for toxicity/bullying
- University-specific feeds (DU, BUET, NSU, etc.)
- Trending topics surface important issues
- Voting system (upvote/downvote)
- AI weekly summary of campus sentiment

Technical Implementation:

- Real-time: Firebase Realtime Database (free)
- Moderation: Perspective API (Google, free)
- Sentiment: HuggingFace transformers
- Trending algorithm: Reddit-style hot ranking
- University verification: Email domain (@du.ac.bd)

Features:

- Post types: Question, Complaint, Announcement, Meme
- Tags: #UnfairGrading #MentalHealth #CourseReview

- Tags: #UnfairGrading #MentalHealth #CourseReview
- Comments (also anonymous)
- Report system → AI reviews → human moderator
- Weekly digest: "Top 5 issues this week at BUET"

Safety Measures:

- AI blocks: hate speech, harassment, explicit content
- Rate limiting: 5 posts/day per user
- Verified students only (no outsiders)
- Report threshold → auto-hide post

User Flow:

1. Student posts: "Professor X is unfair with grading"
2. AI checks for toxicity → Approved
3. Post shows in DU feed
4. Other students upvote/comment
5. Trends if 50+ upvotes in 24 hours
6. AI summary: "23% of posts this week about grading issues"

Business Value:

- Daily engagement (check feed multiple times)
- Community building
- Universities pay for sentiment data
- Advertisers target students by campus

FEATURE 8: PEER TUTORING MARKETPLACE

Problem: Professional tutors expensive, hard to find peer help
Solution:

- Senior students offer tutoring
- Junior students book sessions
- AI matches based on learning style
- In-platform video calls (WebRTC)
- Payment: bKash/Nagad/Bank
- Review/rating system
- Platform takes 10-15% commission

Technical Implementation:

- Video: WebRTC (simple-peer library, free)
- Payment: SSLCommerz integration

- Payment: SSLCommerz integration
- Matching: Cosine similarity algorithm
- Scheduling: Calendar integration
- Reviews: 5-star rating + text

Features:

- Tutors set: Subject, hourly rate, availability
- Students search: "Math tutor for calculus"
- Book: 1-hour session for 200 taka
- Video call in-app (no Zoom needed)
- Auto-record (optional) for review
- Refund if student dissatisfied

Tutor Requirements:

- Must have completed course with A grade
- Verification: Upload transcript
- Initial sessions at 50% rate (build reputation)
- Top tutors get "Verified Expert" badge

Student Benefits:

- Much cheaper than private tutors (150-300 taka vs 1000+ taka)
- On-demand (book session today)
- Peer teaching often more relatable
- Try before committing (first 10 min free)

Business Model:

- Platform fee: 15% per transaction
- Example: Student pays 200 taka
    - Tutor receives: 170 taka
    - Platform keeps: 30 taka
- If 100 sessions/month: 3,000 taka revenue
- Scale to 1000 sessions/month: 30,000 taka

User Flow:

1. Student: "I need help with calculus integration"
2. Platform shows 5 tutors (ratings, rates, availability)
3. Student books Tutor A (250 taka/hour, 4.8 stars)
4. Tutor confirms
5. At scheduled time → video call starts
6. Session ends → student pays via bKash

7. Both rate each other

Business Value:

- Recurring revenue (students book multiple times)
- Network effects (more tutors = more students)
- Low marginal cost (platform just facilitates)
- Premium: Priority matching, lower fees

FEATURE 9: FINANCIAL LITERACY DASHBOARD

Problem: Students terrible with money, no budgeting skills
Solution:

- Track expenses (manual entry or SMS parsing)
- AI categorizes spending (food, transport, etc.)
- Shows spending patterns with charts
- Budget suggestions based on income
- Savings goals with progress tracking
- Educational content (compound interest, investing)
- Bangladesh-specific (bKash, Nagad integration)

Technical Implementation:

- Database: PostgreSQL for transactions
- Charts: Chart.js for visualization
- AI categorization: Simple keyword matching + ML
- SMS parsing: If user shares bKash SMS (optional)
- Educational content: Curated articles/videos

Features:

- Manual entry: "Spent 150 taka on lunch"
- Auto-categorize: AI knows "lunch" = Food
- Monthly reports:
    - Food: 3,000 taka (40%)
    - Transport: 1,500 taka (20%)
    - Entertainment: 1,000 taka (13%)
    - Other: 2,000 taka (27%)
- Budget alerts: "You've spent 80% of food budget"
- Savings challenges: "Save 500 taka this week"

Educational Modules:

- Week 1: Budgeting basics
- Week 2: Understanding interest
- Week 3: Emergency funds
- Week 4: Simple investments (savings accounts)
- Week 5: Financial goals

User Flow:

1. Student inputs monthly allowance: 7,500 taka
2. AI suggests: 3,000 food, 1,500 transport, 1,000 fun, 2,000 savings
3. Student logs expenses daily
4. Mid-month alert: "You're overspending on food by 500 taka"
5. AI tip: "Cook at home 2x/week to save 600 taka/month"
6. End of month: Report shows savings: 1,800 taka
7. Gamification: "Saved for 3 months straight! 🏆"

Business Value:

- Financial services partnerships (banks, fintech)
- Affiliate: Recommend savings accounts, investment apps
- Premium: Advanced analytics, investment tracking
- B2B: Parents pay to monitor student spending

FEATURE 10: AI STUDY COMPANION (ENHANCED)

Not just chat - actually useful study features:

- Pomodoro timer with AI-optimized breaks
- Study music player (lo-fi beats)
- Quiz generator from PDFs/notes
- Deadline tracker with smart reminders
- Focus mode (blocks distracting sites)
- Voice study buddy (talk through problems)
- Daily motivation + study tips

Technical Implementation:

- Timer: JavaScript setInterval
- Music: Royalty-free lo-fi tracks or simple beat generator
- Quiz gen: Extract text from PDF, AI creates questions
- Focus mode: Browser extension (Chrome/Firefox)
- Voice: Web Speech API

- Chat: Ollama Llama 3.1 with RAG

Features:

- Smart timer: "You focused 2 hours, take 20 min break"
- Focus score: Tracks distractions, gives daily score
- Quiz yourself: Upload chapter PDF → AI makes 10 questions
- Voice mode: "Explain photosynthesis to me like I'm 10"
- Deadline panic: "Exam in 3 days, here's your crash course plan"

AI Chat Capabilities:

- Explain concepts simply
- Create study plans
- Review your notes and suggest improvements
- Generate practice problems
- Motivational support
- Study technique recommendations

User Flow:

1. Student: "I have a physics exam in 5 days, I'm not prepared"
2. AI: "Let's create a crash course plan. What topics?"
3. Student: "Mechanics, thermodynamics, optics"
4. AI: Creates day-by-day schedule
    - Day 1: Mechanics basics (2 hours)
    - Day 2: Mechanics problems (3 hours)
    - Day 3: Thermodynamics (3 hours)
    - Day 4: Optics (2 hours)
    - Day 5: Full review + practice test
5. Tracks progress daily
6. Adjusts plan if student falls behind

Business Value:

- Core engagement feature
- Students use daily
- Differentiation: More than just chat
- Premium: Unlimited AI queries

==========

PART 2: TECHNICAL ARCHITECTURE

TECH STACK (ALL FREE FOR MVP)

FRONTEND:

- React (Vite for build)
- Tailwind CSS for styling
- shadcn/ui for components
- Chart.js for visualizations
- Socket.io-client for real-time
- React Router for navigation
- Axios for API calls

Hosting: Vercel (free tier)

BACKEND:

- Python 3.11+
- FastAPI (REST API)
- SQLAlchemy (ORM)
- Alembic (migrations)
- Pydantic (validation)
- python-jose (JWT auth)

Hosting: Railway (free tier)

DATABASE:

- PostgreSQL (main database)
- Redis (caching)
- Chroma (vector database for RAG)

Hosting: Supabase (free tier includes Postgres)

AI/ML:

- Ollama (run Llama 3.1 locally - FREE)
- Groq API (free tier - 30 req/min)
- HuggingFace Transformers (free models)
- spaCy (NLP)
- Tesseract (OCR)

REAL-TIME:

- Socket.io (WebSocket)
- Firebase Realtime Database (free tier)

STORAGE:

- Supabase Storage (free tier - 1GB)
- For PDFs, images, documents

AUTHENTICATION:

- Supabase Auth (free)
- JWT tokens
- Google OAuth

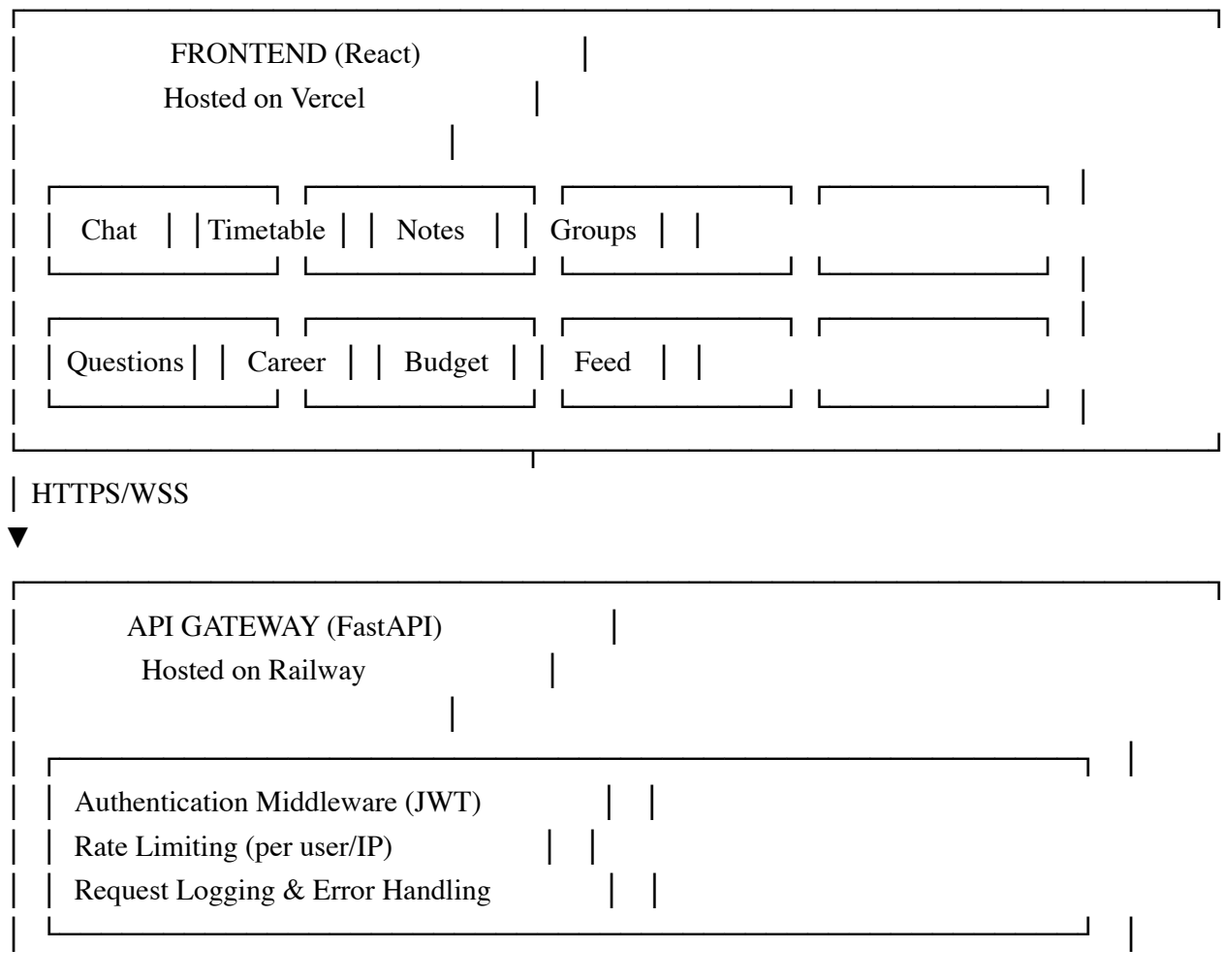
EXTERNAL APIs (ALL FREE):

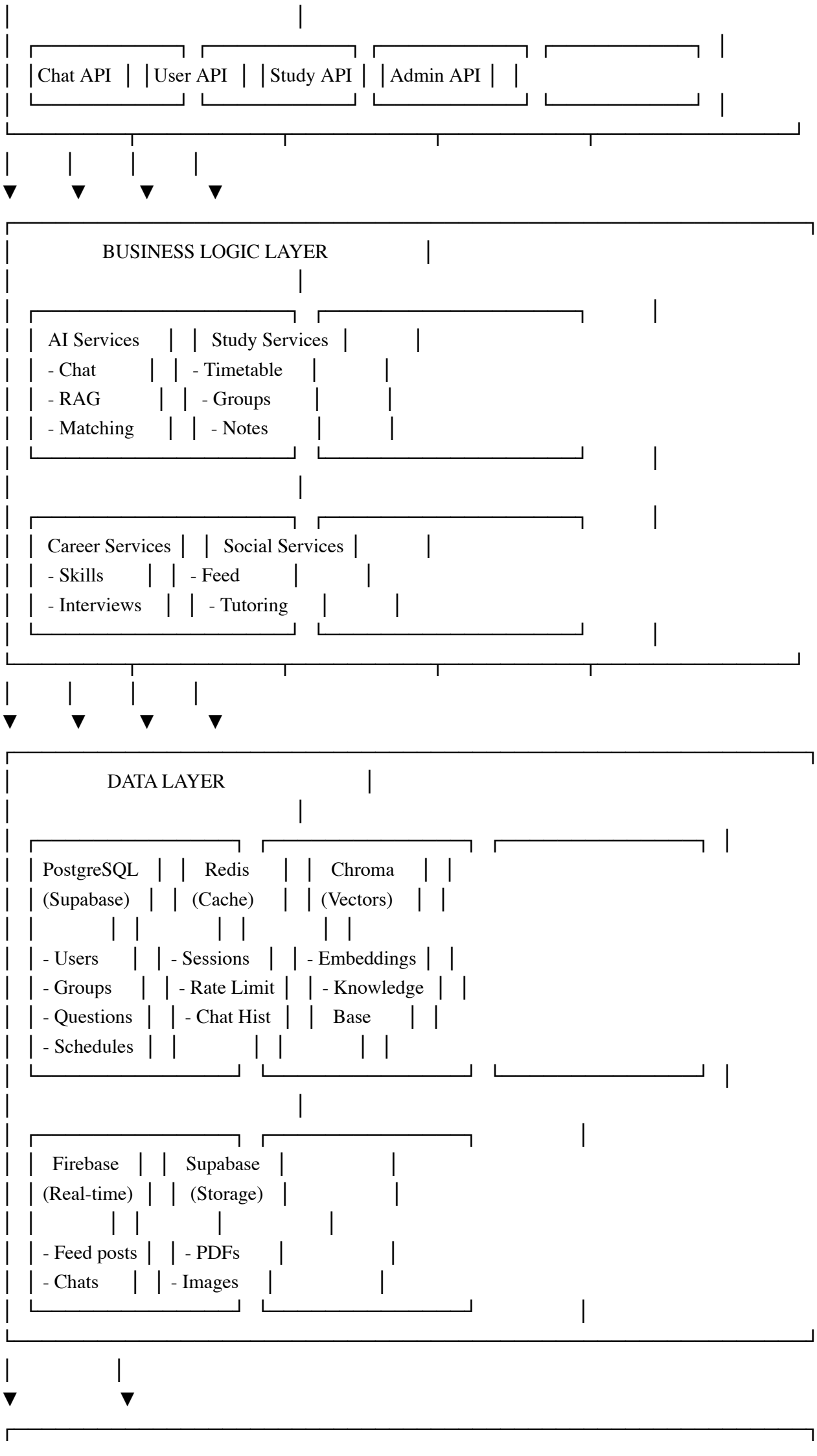
- Google Calendar API
- Perspective API (content moderation)
- Web Speech API (voice)
- Groq (fast LLM inference)

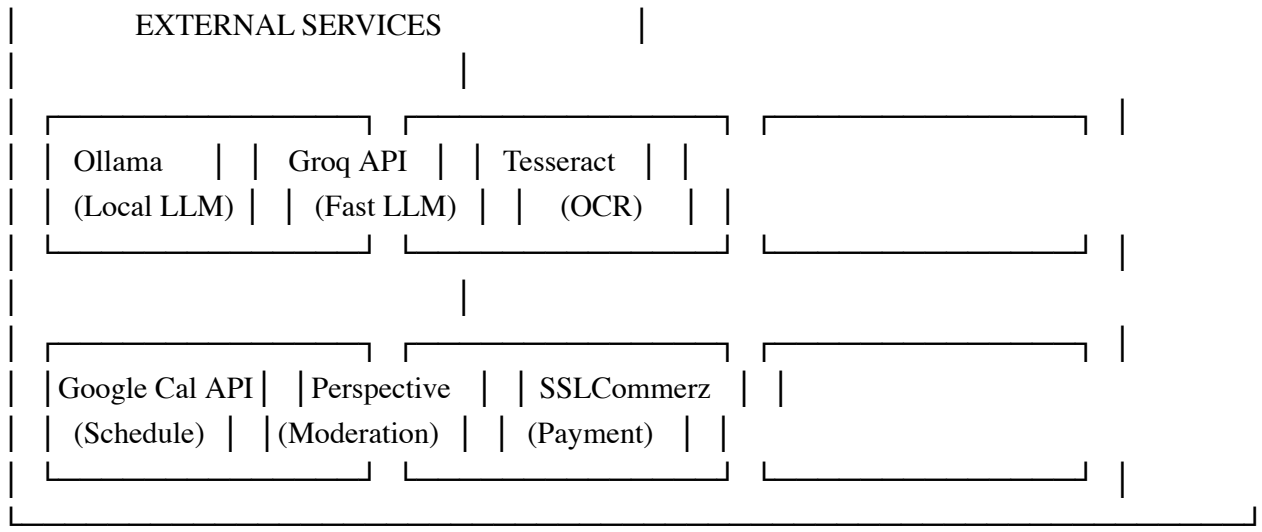
PAYMENT (For Tutoring):

- SSLCommerz (Bangladesh)
- bKash Merchant API
- Nagad (later)

SYSTEM ARCHITECTURE DIAGRAM




Chat API
User API
Study API
Admin API
BUSINESS LOGIC LAYER
AI Services
- Chat
- RAG
- Matching
Study Services
- Timetable
- Groups
- Notes
Career Services
- Skills
- Interviews
Social Services
- Feed
- Tutoring
DATA LAYER
PostgreSQL
(Supabase)
- Users
- Groups
- Questions
- Schedules
Redis
(Cache)
- Sessions
- Rate Limit
- Chat Hist
Chroma
(Vectors)
- Embeddings
- Knowledge
Base
Firebase
(Real-time)
- Feed posts
- Chats
Supabase
(Storage)
- PDFs
- Images



DATABASE SCHEMA (PostgreSQL)

TABLE: users

- id (UUID, primary key)
- email (VARCHAR, unique)
- password_hash (VARCHAR)
- name (VARCHAR)
- university (VARCHAR)
- department (VARCHAR)
- year (INTEGER)
- created_at (TIMESTAMP)
- last_login (TIMESTAMP)
- is_premium (BOOLEAN)
- study_preferences (JSONB)
    - learning_style: visual/auditory/kinesthetic
    - study_hours_per_day: 4
    - preferred_times: ["morning", "evening"]
    - subjects: ["Math", "Physics", "CS"]

TABLE: study_groups

- id (UUID, primary key)
- name (VARCHAR)
- subject (VARCHAR)
- description (TEXT)
- created_by (UUID, foreign key → users.id)
- max_members (INTEGER)
- meeting_schedule (JSONB)
- created_at (TIMESTAMP)

TABLE: group_members

- group_id (UUID, foreign key → study_groups.id)
- user_id (UUID, foreign key → users.id)
- joined_at (TIMESTAMP)
- role (ENUM: member, moderator, admin)

TABLE: questions

- id (UUID, primary key)
- uploaded_by (UUID, foreign key → users.id)
- subject (VARCHAR)
- topic (VARCHAR)
- difficulty (ENUM: easy, medium, hard)
- question_text (TEXT)
- image_url (VARCHAR, nullable)
- answer_text (TEXT, nullable)
- created_at (TIMESTAMP)
- upvotes (INTEGER)
- views (INTEGER)

TABLE: question_answers

- id (UUID, primary key)
- question_id (UUID, foreign key → questions.id)
- user_id (UUID, foreign key → users.id)
- answer_text (TEXT)
- is_ai_generated (BOOLEAN)
- upvotes (INTEGER)
- created_at (TIMESTAMP)

TABLE: schedules

- id (UUID, primary key)
- user_id (UUID, foreign key → users.id)
- google_calendar_id (VARCHAR, nullable)
- events (JSONB)
    - [{ title: "Physics Class", start: "2024-01-16T09:00:00", end: "2024-01-16T10:30:00", type: "class/study/exam", priority: 1-5 }]
- last_synced (TIMESTAMP)

TABLE: notes

- id (UUID, primary key)

- user_id (UUID, foreign key → users.id)
- title (VARCHAR)
- subject (VARCHAR)
- content_text (TEXT)
- original_image_url (VARCHAR, nullable)
- created_at (TIMESTAMP)
- last_edited (TIMESTAMP)
- is_shared (BOOLEAN)

TABLE: flashcards

- id (UUID, primary key)
- note_id (UUID, foreign key → notes.id)
- question (TEXT)
- answer (TEXT)
- next_review (TIMESTAMP)
- ease_factor (FLOAT)

TABLE: campus_feed_posts

- id (UUID, primary key)
- user_id (UUID, foreign key → users.id)
- university (VARCHAR)
- content (TEXT)
- post_type (ENUM: question, complaint, announcement, meme)
- tags (ARRAY of VARCHAR)
- upvotes (INTEGER)
- downvotes (