Unigeek Platform Requirements Document

Below is a high-level requirements document for your EdTech platform, without prescribing specific databases, API endpoints, or implementation methods. It focuses purely on what needs to happen and why, leaving it up to the developer to decide how to implement these requirements.

1. Purpose & Scope

- This document outlines the **functional requirements** for an EdTech platform that offers:
    1. **Course Enrollment**
    2. **Batch Assignments** (system-driven, not user-chosen)
    3. **Payment Processing**
    4. **Live Lecture Management**
    5. **Profile Management** (students & teachers)
    6. **Referral System**
- **Goal**: Provide a seamless experience where students can sign up, purchase courses, be assigned to a batch, attend live lectures, and refer others—while maintaining secure, role-based access.

---

2. User Roles & Access

1. **Student**
    - Must be able to register, sign in, and update their profile.
    - Can view and purchase courses.
    - Cannot choose specific batches; the system automatically assigns them.
    - Gains access to the assigned batch's lectures and resources **only** after successful payment.
    - May refer friends for potential referral bonuses.
2. **Teacher**
    - Registers or is created by an admin.
    - Manages course content, schedules live lectures, and updates lecture info.
    - Updates their own profile (bio, expertise, etc.).
    - No direct involvement in payments, but can see which students are enrolled in their batches.
3. **Admin (or System Administrator)**
    - Full control over creating, updating, and deleting courses.

- Maintains multiple batches for each course (setting capacity, dates, statuses).
- Oversees enrollment records (verifies who is enrolled in which course/batch).
- Manages referral payouts or credits.

---

3. Authentication & Basic Security

1. **User Accounts**
   - Everyone must have a unique identifier (e.g., email or phone) for login.
   - Students and teachers can update their own personal details, but not each other's.
2. **Role-Based Permissions**
   - Students cannot perform teacher or admin tasks (e.g., cannot create courses).
   - Teachers can create/edit lectures within their assigned courses/batches.
   - Admins can perform all actions, including course/batch management.
3. **Session Management**
   - The platform must ensure secure logins and should expire sessions or tokens after a certain duration.

---

4. Course & Batch Requirements

1. **Course Management** (Admin Only)
   - Create new courses with basic details: name, price, duration, description, etc.
   - Update or delete existing courses.
   - List courses for public or internal reference.
2. **Batch Management** (Admin Only)
   - Each course may have multiple batches (e.g., "Jan 2025 Batch," "Summer 2025 Batch").
   - For each batch: set capacity, start/end dates, status ("upcoming," "ongoing," "completed"), etc.
   - The system automatically assigns students to a batch once they pay for a specific course.
   - The admin can manually move students to different batches if needed.
3. **Batch Capacity**
   - System must track how many seats are taken in a batch. If capacity is reached, the admin can decide to create new batches or manage overflows.

---

5. Enrollment & Payment

1. **Course Selection**
    - Students pick a course (they do **not** pick a batch).
    - The platform then determines which batch they will join (e.g., the next upcoming batch).
2. **Payment**
    - Once the course is selected, the student proceeds to payment for that course.
    - The system tracks payment statuses (e.g., "initiated," "successful," "failed").
    - Upon successful payment, the student is officially enrolled in the assigned batch.
3. **Enrollment Records**
    - Each enrollment must link a student to a specific course and batch.
    - Students can only see and access content/lectures for the batches they are enrolled in.
4. **Security**
    - A paid student must not be able to access other batches or courses they haven't purchased.
    - Unauthorized attempts to access other batches/courses should be blocked.

---

6. Live Lectures

1. **Scheduling**
    - Teachers (or admins) can schedule live lectures for a specific batch, setting date/time/duration.
    - Only students enrolled in that batch should view or join the lecture.
2. **Modification**
    - Teachers may add or remove lectures as needed (e.g., if they want to reschedule).
    - Admins can override or manage as well.
3. **Access Control**
    - The system verifies that a student's batch matches the lecture's batch before granting access.
    - If a student or teacher tries to view a lecture outside their batch, the system must deny access.

---

7. Profile Management

1. **Student Profile**

    - Fields: name, email, phone, address, etc.
    - Optionally, store images like a profile photo.
    - Students must be able to update these details securely.
2. **Teacher Profile**
    - Fields: name, email, phone, expertise, bio, profile photo, etc.
    - Teachers manage their own data.
3. **Privacy & Ownership**
    - A user can only update their own profile (unless an admin overrides for some specific reason).

---

8. Referrals

1. **Referral Form**
    - Current students can input a friend's info (name, email, phone, etc.) for referral.
    - The system links the referral to the current student.
2. **Referral Tracking**
    - If the referred friend signs up and enrolls, the platform must track that relationship.
    - A bonus or credit is assigned to the referrer student if the new student successfully pays for a course.
3. **Visibility & Commission**
    - Students should be able to see a summary of their referral status (whether their friend has signed up, paid, etc.).
    - Admin can see all referral data and manage payouts.

---

9. Additional Requirements & Constraints

1. **Scalability**
    - The system should handle multiple concurrent students, courses, and teachers without performance degradation.
2. **Audit Trails**
    - Recommended to keep logs of enrollment changes, payment updates, and referral signups for later analysis or dispute resolution.
3. **Notifications** (Optional/Recommended)

- Email or SMS for key events: enrollment success, payment confirmation, lecture schedule changes, etc.

4. **Internationalization (i18n)** (Optional)
    - Consider storing text in a way that could be localized in multiple languages if needed.

---

10. Conclusion

These **functional requirements** provide a big-picture view of what needs to be built:

1. **Secure authentication** with clear role distinctions (student, teacher, admin).
2. **Automatic batch assignment** after a student pays for a course.
3. **Live lectures** tied to a specific batch with controlled access.
4. **Referral system** that rewards existing students.
5. **Profile management** for both students and teachers.

Developers should now determine the **specific database design** (tables, schemas) and **API endpoints** (HTTP verbs, paths, request/response models) that satisfy these requirements. The implementation approach—Node.js with a relational or NoSQL database, or any other tech stack—remains at their discretion, as long as it fulfills the business logic and security standards described here.