Duolingo for police statal exams

Introduction

In this document we will explain in detail the design, requirements and goals for the development of an app that uses a gamified learning style as duolingo for the police statal exam in spain. We aim to create in less than two months the first iteration of the app which consists in three keypoints:

- **Database**: Creating a dataset with the exam questions and a database accessible by the backend to server questions to the frontend. The database will also have to store the information for the users and everything needed for the app.
- **App**: The app is divided in two keypoints:
  - **Frontend**: The appearance of the app, a modern and intuitive UI that makes learning fun and easy, an app with a strong social component with laderboards, leagues and friendly matches, a personally focused app with tailored sets of lessons based on the users weak and strong points, an app that encourages personal development with extra lesson challenges as, how many questions without failing in a row you can do, or lightning rounds.
  - **Backend**: A robust server that will serve the questions to the frontend, the information for the social activity and will carry all the logic and data management needed.
- **Learning path**: The order, quantity and frequency of questions delivered to the user, the learning experience, the types of lessons, etc. Will be designed by an expert in the field so the learning curve is smooth and ensures enjoyable and useful learning. The expert will decide the content of the lessons, their order and some parameters of the challenges.

This three keypoints will interact with each other's, the interaction between the app and the database is obvious but a deeper introduction for the other interactions must be said:

- **Database – Learning path interaction:** The learning expert that will design the lessons is not computer fluent, and the questions and syllabus of this type of exams is quite volatile, so the learning expert will need to edit, add and delete questions from the database at any moment. As he will not learn how to use sql, a learning expert dashboard must be created apart of the app. A website with authentication that will be served by the server where the learning expert can see, edit add and delete questions from the database easily.
- **App – Learning path interaction:** The app will only be a lesson delivery interface, the structure of a lesson will always be the same, a set of ordered sessions and a session is a set of random questions selected with certain parameters. The lessons are not going to be hardcoded, as we said earlier the syllabus is volatile and the learning expert needs to be able to modify the learning path easily. For this reason the lessons should be created, edited and deleted from the dashboard also, saved in the database and then the app will server them in the stipulated order. So the app will only be an skeleton ready to server lessons and the early mentioned challenges, which will have an algorithmic behavior but the parameters of that algorithms also have to be editable by the learning expert.

Database

Learning path tables

The database will be created in supabase for a clean and fast setup SQL.

The syllabus is organized by this hierarchy:

1) Syllabus
    a) Blocks
        i) Topics
            (1) Headings
                (a) Concepts
                    (i) Questions

Each question belongs to a concept, that belongs to a heading, that belongs to a topic, that belongs to a block.

The information needed for blocks, topics, headings and concepts is:

- Name
- Description: nullable
- Order/Position: An integer to define the sequence in which blocks, topics, headings, and concepts appear. This is critical since your hierarchy is ordered and the learning path needs to present content in a specific sequence.
- Status/Visibility: A boolean or enum field (e.g., active, draft, archived) to control whether content is visible to users. This allows the learning expert to prepare content without publishing it immediately.
- Foreign key to their parent

The information needed for questions is:

- Text: Question text
- Option A
- Option B
- Option C: These exams only have three possible answers" (clearer phrasing)
- Correct option: Has to be a, b or c
- Explanation: A concise and easy explanation to understand the answer
- Difficulty: An integer number that will represent the difficulty of the question
- Source: From what source was the question extracted.

For the lessons, the learning expert still has to design their requirements but for now on they need this information:

- Name
- Order/Position: An integer defining the lesson sequence in the learning path.
- Prerequisites: nullable - References to other lessons that must be completed first. Enables skill tree/dependency logic.
- XP reward: Integer - Gamification element defining experience points earned upon completion.

- Status: enum (active, draft, archived) - Allows the learning expert to prepare lessons without making them live.

For the sessions, the learning expert still has to design their requirements but for now on they need this information:

- Name
- Foreign key to the lesson it belongs
- Pool of questions: A list of question IDs that can be served in this session. These questions will be sampled according to the question selection strategy. Not all questions in the pool will be shown in a single attempt.
- Number of questions: Number of questions to be sampled from the pool
- Order/Position: An integer defining the session order within the lesson.
- Question selection strategy: enum (random, weighted_by_difficulty, adaptive, spaced_repetition) - Defines how questions are sampled from the pool. Even if you start with "random", this parameter lets the learning expert experiment with different strategies later.

Note: Sessions need to track which specific questions were shown to each user and their responses. This requires a many-to-many relationship table between users, sessions, and questions with fields for:

- Timestamp
- User answer
- Correct/incorrect
- Time taken

This data is essential for the "tailored sets of lessons based on the users weak and strong points" mentioned in your introduction and for spaced repetition algorithms.

For the learning path global configuration, the following parameters are needed:

- Retry penalty: Boolean or XP amount - Whether retrying a session affects XP rewards
- Spaced repetition intervals: Array of integers (days) - Defines when questions should be re-presented

These parameters can be edited by the learning expert from the dashboard without changing the app code.

For the algorithmic challenges available in the app (lightning rounds, review sessions, etc.):

- Challenge Template ID: Primary key
- Challenge name: Display name (e.g., "Lightning Round", "Review Weak Points", "Perfect Streak Challenge")
- Challenge type: enum (lightning_round, review_weak_topics, review_mistakes, speed_run, accuracy_challenge, spaced_repetition_review) - Category
- Description: What this challenge does and when it's available
- Icon URL: Challenge icon for UI
- Time limit: Integer (seconds) - nullable for untimed challenges
- Number of questions: Integer - How many questions to serve
- Question selection algorithm: enum (random, weakest_topics, recent_mistakes, spaced_repetition, difficulty_ascending, high_frequency) - How questions are chosen

- Scoring formula: enum (standard, time_bonus, combo_multiplier, no_penalty) - How score is calculated
- XP multiplier: Float - Bonus multiplier for XP rewards (e.g., 1.5x for lightning rounds)
- Unlock criteria: JSON - nullable, conditions to access this challenge (e.g., {"min_lessons_completed": 5} or {"min_level": 3})
- Cooldown period: Integer (hours) - nullable, time before challenge can be replayed
- Active: Boolean - Whether this challenge is currently available
- Order/Position: Integer - Display order in challenges menu

App logic tables

Tables that will be needed for the app functionality like users profiles, history, gamification mechanics, social features, and challenges.

Users

The core user profile information:

- User ID: Primary key, unique identifier
- Username: Unique, for display in leaderboards and social features
- Email: Unique, for authentication
- Password hash: Encrypted password storage
- Full name: nullable - Optional real name
- Profile picture URL: nullable - Avatar/photo
- Date joined: Timestamp of account creation
- Last active: Timestamp of last app usage
- Preferred study time: nullable - Time of day for notification reminders (e.g., "18:00")
- Daily goal: Integer - Target XP or lessons per day (default: configurable globally)
- Notification preferences: JSON or boolean fields - Controls for streak reminders, friend activity, league updates
- Account status: enum (active, suspended, deleted) - Account state

User progress

Tracks individual user advancement through the learning path:

- User ID: Foreign key to Users
- Lesson ID: Foreign key to Lessons
- Status: enum (not_started, in_progress, completed, locked) - Current lesson state
- Current session: Integer - Which session within the lesson the user is on
- Completion percentage: Integer (0-100) - Progress within the lesson
- Started at: Timestamp - nullable, when user first opened this lesson
- Completed at: Timestamp - nullable, when user finished this lesson
- XP earned: Integer - XP gained from this lesson
- Stars earned: Integer (0-3) - Performance rating for this lesson
- Best score: Integer (percentage) - Highest score achieved across all attempts
- Attempts: Integer - Number of times lesson was attempted

User Session History

Detailed record of each session attempt (this is the many-to-many table mentioned in Learning path section):

- History ID: Primary key
- User ID: Foreign key to Users
- Session ID: Foreign key to Sessions
- Lesson ID: Foreign key to Lessons
- Started at: Timestamp
- Completed at: Timestamp - nullable if session not finished
- Questions shown: Array of question IDs - Exact questions presented in this attempt
- User answers: Array of user responses - Corresponds to questions shown
- Correct answers: Array of booleans - Corresponds to questions shown
- Time per question: Array of integers (seconds) - Time taken per question
- Total score: Integer (percentage) - Overall session performance
- XP earned: Integer - XP from this session
- Passed: Boolean - Whether user met minimum passing score
- Question selection strategy used: enum - Which strategy was applied for this session

User Gamification Stats

Tracks gamification metrics per user:

- User ID: Foreign key to Users (Primary key)
- Total XP: Integer - Lifetime experience points
- Current level: Integer - Calculated from total XP
- XP to next level: Integer - Points needed to level up
- Current streak: Integer (days) - Consecutive days of activity
- Longest streak: Integer (days) - Best streak ever achieved
- Last streak date: Date - Last day streak was maintained
- Streak freeze count: Integer - Number of streak freezes available (power-up that protects streak)
- Total lessons completed: Integer
- Total questions answered: Integer
- Total correct answers: Integer
- Accuracy rate: Float (percentage) - Overall correctness
- Current league: enum (bronze, silver, gold, diamond, obsidian) - League tier
- League position: Integer - Rank within current league
- League points this week: Integer - XP earned in current week (resets weekly)
- Achievements unlocked: Array or count - Number/list of badges earned
- Lightning round high score: Integer - Best performance in lightning challenge
- Perfect streak record: Integer - Most consecutive correct answers without a mistake

Achievements/Badges

Defines available achievements in the system:

- Achievement ID: Primary key
- Name: Achievement title (e.g., "First Steps", "Week Warrior", "Perfect 10")
- Description: What the achievement represents
- Icon URL: Badge image
- Type: enum (streak, completion, accuracy, challenge, social, special) - Category
- Unlock criteria: JSON - Conditions to earn this badge (e.g., {"streak_days": 7} or {"lessons_completed": 10})

- XP reward: Integer - Bonus XP for unlocking
- Rarity: enum (common, rare, epic, legendary) - Badge tier
- Order/Position: Integer - Display order in achievement list
- Active: Boolean - Whether this achievement is currently earnable

User Achievements

Tracks which achievements each user has unlocked:

- User ID: Foreign key to Users
- Achievement ID: Foreign key to Achievements
- Unlocked at: Timestamp
- Shown notification: Boolean - Whether user has seen the unlock notification

Leagues

Weekly competitive leagues:

- League ID: Primary key
- League name: enum (bronze, silver, gold, diamond, obsidian)
- Start date: Timestamp - When this league week began
- End date: Timestamp - When this league week ends
- Promotion threshold: Integer - Minimum rank to move up a league
- Demotion threshold: Integer - Maximum rank to avoid moving down
- Active: Boolean - Whether this is the current active league period

League Participants

Tracks users in each league period:

- Participant ID: Primary key
- User ID: Foreign key to Users
- League ID: Foreign key to Leagues
- XP earned this week: Integer - Points accumulated during league period
- Current rank: Integer - Position in league leaderboard
- Previous rank: Integer - Rank in last update (for showing movement)
- Promoted: Boolean - nullable, set at end of week
- Demoted: Boolean - nullable, set at end of week

Friends

Social connections between users:

- Friendship ID: Primary key
- User ID 1: Foreign key to Users
- User ID 2: Foreign key to Users
- Status: enum (pending, accepted, blocked) - Friendship state
- Requested by: Foreign key to Users - Who initiated the friend request
- Created at: Timestamp
- Accepted at: Timestamp – nullable

Friendly Matches

Head-to-head challenges between friends:

- Match ID: Primary key

- Challenger ID: Foreign key to Users
- Opponent ID: Foreign key to Users
- Session ID: Foreign key to Sessions - Which session to compete on
- Status: enum (pending, in_progress, completed, declined) - Match state
- Created at: Timestamp
- Started at: Timestamp - nullable
- Completed at: Timestamp - nullable
- Challenger score: Integer - nullable until completed
- Opponent score: Integer - nullable until completed
- Winner ID: Foreign key to Users - nullable until completed
- XP reward: Integer - Bonus XP for winner

Challenge History

Records for special algorithmic challenges (lightning rounds, perfect streaks, etc.):

- Challenge ID: Primary key
- User ID: Foreign key to Users
- Challenge type: enum (lightning_round, perfect_streak, accuracy_challenge, speed_run) - Challenge category
- Started at: Timestamp
- Completed at: Timestamp - nullable
- Questions answered: Integer
- Correct answers: Integer
- Time taken: Integer (seconds)
- Score: Integer - Challenge-specific scoring
- XP earned: Integer
- New personal best: Boolean - Whether this beat user's previous record

Notifications

System and social notifications for users:

- Notification ID: Primary key
- User ID: Foreign key to Users
- Type: enum (streak_reminder, friend_request, league_update, achievement_unlock, match_challenge, level_up) - Notification category
- Title: Notification heading
- Message: Notification body text
- Related entity ID: Integer - nullable, reference to related item (e.g., friend request ID, match ID)
- Related entity type: enum - nullable, type of related entity
- Created at: Timestamp
- Read at: Timestamp - nullable
- Action URL: nullable - Deep link within app (e.g., "app://matches/123")

Daily Activity Log

Tracks daily user engagement for streak calculation and analytics:

- Log ID: Primary key
- User ID: Foreign key to Users

- Activity date: Date
- Sessions completed: Integer
- Lessons completed: Integer
- Questions answered: Integer
- Correct answers: Integer
- XP earned: Integer
- Time spent: Integer (minutes) - Total active time
- Streak maintained: Boolean - Whether activity counted toward streak

App Configuration

Global settings editable by learning expert from dashboard:

- Config key: Primary key, unique identifier (e.g., "daily_xp_goal_default")
- Config value: Text or JSON - The actual value
- Data type: enum (integer, boolean, string, json, array) - Value type
- Description: What this configuration controls
- Category: enum (gamification, learning, social, challenges, notifications) - Grouping
- Last updated: Timestamp
- Updated by: Foreign key to admin/expert user who made the change