**Core Features of "Shifter"**

1. **Daily MCQs**: Users get a new set of questions daily based on selected topics.
2. **Progress Tracking**: Track scores, streaks, and performance over time.
3. **Topic Selection**: Users choose from a variety of topics or categories.
4. **Gamification**: Badges, leaderboards, and streaks to keep users engaged.
5. **Feedback**: Explanations for correct and incorrect answers.
6. **Notifications**: Daily reminders to complete quizzes.

**Backend & API**

1. **Backend Stack**:
   * Use **Django REST Framework** or **Express.js** for building a scalable backend.
   * **PostgreSQL** or **Firebase Firestore** for the database.
2. **API Endpoints**:
   * GET /questions: Fetch daily questions.
   * POST /progress: Submit user answers and update progress.
   * GET /progress: Fetch user progress and streaks.
   * GET /topics: Fetch available topics.
3. **Authentication**:
   * Use Firebase Authentication (easy integration with Flutter) for email/password and social login.

**Flutter Development**

**1. Folder Structure**

Organize your app for scalability and readability:

vbnet

CopyEdit

lib/

models/

services/

providers/

screens/

widgets/

utils/

**2. Key Flutter Packages**

* **State Management**:
  + Use provider, riverpod, or bloc for state management.
* **Networking**:
  + http or dio for API calls.
* **Database**:
  + Use sqflite for local offline storage or Hive for lightweight NoSQL.
* **Notifications**:
  + firebase\_messaging for push notifications.
* **Animations**:
  + flutter\_animate or rive for creating engaging transitions and effects.

**3. UI Design**

* **Homepage**: Show a dashboard with:
  + Today's topics and progress.
  + A streak tracker.
* **Quiz Screen**:
  + A clean interface for MCQs with options, a timer, and a "Next" button.
* **Progress Page**:
  + Display graphs using fl\_chart to visualize growth.
* **Settings Page**:
  + Manage notifications, theme (light/dark mode), and account details.

**Additional Features**

1. **Offline Mode**:
   * Preload daily questions so users can attempt quizzes without internet.
2. **Dark Mode**:
   * Use Flutter’s built-in ThemeData for light/dark mode toggling.
3. **Localization**:
   * Support multiple languages using flutter\_localizations.

**CI/CD for Flutter**

1. **Version Control**:
   * Use GitHub or GitLab for version control.
2. **CI/CD Tools**:
   * **Codemagic**: Tailored for Flutter projects.
   * **GitHub Actions**: Automate testing and deployments.
3. **Testing**:
   * Write **unit tests** for business logic using flutter\_test.
   * Implement **widget tests** for UI components.