هيكل بناء موقع إدارة جيم — خريطة تفصيلية

مرحبًا بحر — عملت لك خريطة بناء مفصّلة جاهزة تقدر تبدأ عليها فورًا. هذي الوثيقة تحتوي على: صفحات (Routes)، مكوّنات الواجهات، مخطط قاعدة البيانات على (Firestore + Storage) Firebase، صلاحيات، تدفقات الدفع والإشعارات، سير العمل لتجديد وإلغاء الاشتراكات، وملفّات ومجلدات مقترحة لـ HTML/CSS/JS + ربط الصور على Google Drive.

> ملاحظة سريعة: ركّز على التنفيذ خطوة-بخطوة — سأعطي أولويات تنفيذ (MVP) في النهاية حتى تقدر تطلع نسخة شغالة بسرعة.

---

1) نظرة عامة على الأدوار (Roles)

- **admin**: إدارة كاملة (Dashboard، إدارة كباتن، إعدادات الجيم، تقارير، تعديلات يدويّة على الاشتراكات/المشتركين).
- **captain** (كابتن/مدرّب): تسجيل مشتركين جدد، تجديد اشتراكات، تسجيل حصص فردية، عرض وتعديل بيانات المشتركين (حسب الصلاحيات)، رفع سندات الدفع، إرسال إشعارات (حسب التفعيل).
- **trainee** (متدرب): لا يحتاج الآن — سنصمم صفحة بروفايل له لاحقًا.

> كل دخول يتم عبر صفحة Login موحدة مع اختيار نوع الحساب (admin/captain). captains قد يطلب منهم إنشاء طلب تسجيل ويحتاج موافقة admin.

---

2) صفحات (Routes) رئيسية

(ملاحظة: كل صفحة لها مسار واضح واسم ملف مقترح)

1. `/login` — صفحة تسجيل دخول / إنشاء حساب كابتن (login.html / register-captain.html)
2. `/dashboard` — لوحة رئيسية للادمن (admin-dashboard.html)
3. `/captain-home` — صفحة رئيسية للكابتن بعد الدخول (captain-home.html)
4. `/members` — صفحة قائمة المشتركين مع شريط بحث مرن وفلاتر (members.html)
5. `/member/:id` — صفحة عرض / تعديل تفاصيل مشترك (member-detail.html)
6. `/register-member` — صفحة تسجيل مشترك جديد (register-member.html)
7. `/single-session` — تسجيل حصص فردية (single-session.html)
8. `/subscriptions/expired` — اشتراكات منتهية (subscriptions-expired.html)
9. `/subscriptions/renewed` — اشتراكات مجددة (subscriptions-renewed.html)
10. `/attendance` — تسجيل حضور (attendance.html) عبر QR أو يدويًا
11. `/captains` — إدارة حسابات الكباتن وطلبات التسجيل (captains.html)
12. `/training-plans` — نظام انشاء برامج تدريبية/غذائية (future) (training-plans.html)
13. `/reports` — صفحة تحليلات وماليات حسب فترة (reports.html)
14. `/settings` — إعدادات الجيم، طرق دفع مفعلة، API keys (settings.html)

---

3) المكوّنات (Components) UI أساسية

- Header (اسم الجيم + أيقونة القائمة الجانبية + صورة الكابتن (قابلة للنقر لفتح صفحته)).

- Sidebar (قابلة للانزلاق/الإغلاق — تفتح وتُغلق عند الخروج بالماوس أو الضغط مجددًا).
- Members table with: صورة، اسم، هاتف، نوع الاشتراك، تاريخ البداية، تاريخ الانتهاء، باقي الأيام، أزرار (عرض، تعديل، حذف، تجديد، إلغاء/بلوك).
- Member detail card: صورة كبيرة، بيانات الاتصال، تاريخ إنشاء الحساب، من أنشأه، تاريخ كل تجديد (قائمة)، سجلات المدفوعات، زر رفع إثبات الدفع.
- Modal dialogs: تأكيد حذف، تأكيد بلوك مع حقل سبب.
- Form components: date picker، currency input، payment method selector (Cash default)، Upload input for receipt.
- Notifications toggle per-captain (switch) عند تسجيل/تجديد — يتحكم بإرسال رسالة واتس آب.
- QR scanner modal (للـ attendance).

---

4) مخطط قاعدة البيانات — Firestore (اقتراح Collections)

التعريفات التالية تستهدف Firestore (NoSQL). استخدم `createdBy` و`createdAt` في كل مستند لأغراض التدقيق.

Collections

1. **users** (المشغلون: admin & captains)

```
{ id, name, phone, email, role, photoUrl, jobTitle, dob, approved: bool,
permissions: {createPlan: bool, renew: bool, block: bool, ...},
createdAt, createdBy }
```

2. **members** (المشتركين)

```
{ id, name, phone, photoUrl, dob, createdAt, createdBy (captainId),
status: "active"|"expired"|"blocked"|"cancelled", notes }
```

3. **subscriptions**

```
{ id, memberId, type: "normal"|"cardio"|"single", startDate, endDate,
price, paymentMethod: "cash"|"visa"|"instapay", createdAt, createdBy
(captainId), receipts: [storageUrl], renewHistory: [{date, amount,
method, by}], isAutoCash: bool }
```

4. **single_sessions**

```
{ id, name, phone (optional), date, price, createdAt, createdBy,
receiptUrl }
```

5. **attendance**

```
{ id, memberId, dateTime, method: "qr"|"manual", recordedBy (captainId|null) }
```

6. **payments** (سجل مالي موحد)

```
{ id, type: "new_subscription"|"renewal"|"single_session", refId, memberId, amount, method, date, by (captainId), receiptUrl }
```

7. **notifications_log**

```
{ id, memberId, type: "welcome"|"renewal", message, sentAt, via: "whatsapp", toPhone, by (captainId) }
```

8. **audit_logs**

```
{ id, action: "create_member"|"renew"|"block"|"delete", targetId, by, reason?, timestamp }
```

9. **captain_requests** (طلبات التسجيل)

```
{ id, name, phone, email, photoUrl, jobTitle, submittedAt, status: "pending"|"approved"|"rejected", reviewedBy }
```

---

5) Storage & الصور

- استخدم Firebase Storage أو Google Drive links (إذا تفضّل تخزين الصور في Drive، احفظ الروابط في `photoUrl` أو `receipts`).
- تنظيم المجلدات: `/members/{memberId}/photo.jpg`, `/payments/{paymentId}/receipt.jpg`, `/captains/{userId}/photo.jpg`.

---

6) Authentication & Security (مقترح Firebase Auth + Firestore Rules)

- مصادقة عبر **Firebase Auth (email/phone)**.
- عند تسجيل كابتن جديد: يُنشئ حسابًا لكن `approved=false` إلى أن يوافق الادمن.
- قواعد Firestore (مقتطفات):

```
// مثال مُبسّط: فقط الادمن يمكنه حذف members
match /members/{memberId} {
 allow read: if request.auth != null;
 allow create: if request.auth != null && (request.auth.token.role ==
'admin' || request.auth.token.role == 'captain');
 allow delete: if request.auth != null && request.auth.token.role ==
'admin';
}
```

- سجّل `customClaims` على Firebase Auth لتعيين role (admin/captain).

---

7) سير العمل (Flows) — أمثلة مفصّلة

تسجيل مشترك جديد (captain -> register-member)

1. الكابتن يملأ form: صورة، اسم، هاتف، نوع اشتراك، سعر، تاريخ بداية، طريقة دفع (افتراضي Cash).
2. إنشاء مستند `members` + `subscriptions` + سجل `payments` إن كان الدفع فوري.
3. إن كان Notifications مفعّل عند الكابتن: ترسل رسالة واتساب آليًا إلى رقم المشترك (تحتوي: "تم تسجيلك في جيم عبدالودود بتاريخ X .. ينتهي Y"). يُسجَّل هذا في `notifications_log`.

تجديد اشتراك

1. من زر "تجديد" في صفحة المشترك: يفتح modal فيه مدة تجديد (15 يوم/شهر/مخصص) والسعر وطريقة الدفع (Cash default).
2. إنشاء سجل `subscriptions` جديد (أو تحديث endDate) + سجل `payments` + إضافة مدخل في `renewHistory` داخل مستند الاشتراك.
3. إن كانت إشعارات مفعلة -> إرسال واتس.

إرسال إشعارات واتساب

- استخدم خدمة طرف ثالث (WhatsApp Business API أو Twilio). عند نجاح تسجيل/تجديد، استدعي Cloud Function (Firebase Functions) الذي يرسل الرسالة ويسجّلها.

إلغاء اشتراك / بلوك

- عند الضغط على "إلغاء" أو "بلوك" يفتح modal للحصول على السبب. يتم تحديث `subscriptions` / `members.status` وتسجيل entry في `audit_logs`.

تسجيل حضور عن طريق QR

- علّق ملصق يحتوي على رابط مختصر (مثلاً `example.com/qr?mid=<<memberId>>`) أو استخدم رمز QR الذي يقود إلى صفحة تَسجِيل الحضور. عندما يسكن الـ QR، يتم استدعاء endpoint (Cloud Function) الذي يسجل `attendance` تلقائيًا.

---

8) الدفع (Payment flow)

- طرق: Cash (افتراضي), Visa, Instapay (إنستاباي — إن أمكن دمجه عبر بوابة محلية أو تحويل بنكي)

- عند اختيار غير Cash: افتح modal لإتمام عملية الدفع (لـ MVP يمكنك وضع حقل "Upload proof" وانتظار التحقق اليدوي).
- بعد اكتمال الدفع، خزّن `receiptUrl` في `payments` و`subscriptions.receipts`.

---

9) Dashboard ادمن (التقارير)

- فلترة بالتاريخ (from - to). إن لم يتم اختيار فترة، يعرض اليوم الحالي.
- مؤشرات: عدد مشتركين جدد، عدد تجديدات، عدد حصص فردية، إيرادات إجمالية، عدد البلوكات، إلغاءات الاشتراكات، حضور اليوم.
- لكل كابتن: عدد المسجلين، عدد المجدّدين، عدد الملغين، عدد المبلوكين، عدد الحصص المسجلة. زر لعرض تفاصيل كل حدث (مع من أنشأه والسبب إن وُجد).

---

10) صفحة الملف الشخصي للكابتن

- تعرض: صورة، اسم، هاتف، تاريخ الميلاد، وظيفته، إحصاءات: عدد التسجيلات، عدد التجديدات، عدد الحصص الفردية، عدد الإلغاءات، عدد البلوكات التي عملها.
- صلاحية تعديل: يمكنه تعديل صورته، اسمه، هاتفه، تاريخ ميلاده. لا يمكنه تعديل إيميله أو وظيفته.
- زر "تسجيل خروج".

---

11) خواطر تقنية / اقتراحات تنفيذية

- Frontend: HTML + CSS + vanilla JS + Firebase SDK (Firestore, Auth, Storage, Functions). استخدم مكتبة صغيرة لـ UI مثل Tailwind لاحقًا لو حبيت تنسق أسرع.
- Backend: Firebase Functions (Node.js) لعمليات: إرسال WhatsApp، معالجة الدفع، جدولة تنبيهات يومية/تذكير انتهاء اشتراكات.
- Use Cloud Scheduler + Functions لفحص الاشتراكات القريبة للانتهاء وإرسال تنبيهات قبل X يوم.
- لصلاحيات معقدة استخدم `permissions` داخل users وراجعها في الـ Frontend + Security Rules.

---

12) بنية مجلدات مقترحة (project root)

```
/public
  /css
    styles.css
  /js
    auth.js
    dashboard.js
    members.js
    member-detail.js
    payments.js
  /pages
    login.html
    admin-dashboard.html
    captain-home.html
```

```
    members.html
    member-detail.html
    register-member.html
    single-session.html
    attendance.html
  index.html
/firebase
  functions/   (cloud functions: sendWhatsApp, qrAttendance,
scheduledReminders)
  firestore.rules
```

---

13) أمثلة كودية قصيرة (ملاحظات تنفيذية)

- تسجيل مشترك جديد (pseudo-js Firebase):

```
// create member
const memberRef = await db.collection('members').add({name, phone,
photoUrl, createdAt: now, createdBy: uid});
// create subscription
await db.collection('subscriptions').add({memberId: memberRef.id,
startDate, endDate, price, paymentMethod, createdAt: now, createdBy: uid});
// create payment if paid
await db.collection('payments').add({type:'new_subscription', refId:
subscriptionId, memberId: memberRef.id, amount: price, method:
paymentMethod, date: now, by: uid});
```

- إرسال إشعار واتس (via Cloud Function): استدعي خدمة واتساب ثم خزّن سجل في `notifications_log`.

---

14) أولويات التطوير (MVP roadmap)

مرحلة 1 (يوم-5 أيام عمل) - Authentication (admin + captain) + roles (approval flow for captains). - صفحة تسجيل مشترك جديد (form) + حفظ members + subscriptions + payments (أي إثبات رفع ملف). - إرسال إشعار واتس اختياري عند التسجيل (يمكن البداية بـ webhook تجريبي أو تسجيل الرسالة دون إرسال). - Members list + details view + تجديد بسيط.

مرحلة 2 - Dashboard ادمن + تقارير الفترة. - تسجيل حصص فردية، رفع إيصالات. - صفحة الاشتراكات المنتهية والمجددة. - زر بلوك/إلغاء مع audit log.

مرحلة 3 - QR attendance + Cloud Function لسجل الحضور. - دمج بوابة دفع (Visa / Instapay) أو آلية رفع إثبات دفع أو ربط مع موفّر محلي. - System for training plans (AI) — تصميم قاعدة للبيانات وتكامل لاحق.

---

15) نقاط يجب الانتباه لها (Risks & Notes)

- القوانين المحلية للدفع الإلكتروني وWhatsApp API — تحقق من مزود موثوق.
- حماية بيانات المتدربين (أرقام تليفون، تواريخ ميلاد) — اتبع قواعد الخصوصية.
- فكر في إجراء نسخة احتياطية منتظمة لقاعدة البيانات.

---

16) checklist جاهزة للبدء

- [ ] تهيئة مشروع Firebase (Auth, Firestore, Storage, Functions).
- [ ] إعداد customClaims للأدوار.
- [ ] بناء صفحة Login + approval flow للكباتن.
- [ ] بناء صفحة Register Member (MVP) وربطها بFirestore & Storage.
- [ ] تنفيذ صفحة Members list مع بحث وفلترة وبطاقة تفاصيل.
- [ ] إعداد Cloud Function (sendWhatsApp) ودمجه بملف الإعدادات.

إذا بتحب أطلع لك الآن ملف README مفصّل أو ملف JSON لمخطط الـ Firestore أو حتى أمثلة HTML/CSS/JS تشتغل مباشرة على Firebase Hosting أقول أعمله لك — اختَر: `README` أو `Firestore JSON` أو `MVP HTML snippets` وأنا أضبطهم فورًا.

بقولها بصراحة: الخطة دي تكفيك تبني نسخة منتجة بسرعة — لو تريد أسرع نسخة تشغيلية أقدر أطلع لك مشروع MVP مع صفحات HTML/CSS/JS جاهزة لـ Firebase hosting.

خلصت الخريطة — قل لي تاليًا أي حاجة عايز أطلعها لك (README, سكريبت قواعد البيانات، أمثلة صفحات) وابدأ أحضرها لك فورًا.