عنوان :PDF شرح شامل لأكواد الباك-إند(Backend)

مقدمة:

هذا المستند يقدم شرحًا تفصيليًا لأكواد نظام باك-إند متكامل لتطبيق تواصل اجتماعي، يشمل ميزات المصادقة وإدارة الصداقات والدردشة.

.1بنية المشروع:

يتكون المشروع من عدة أقسام رئيسية منظمة كالتالي:

*هـ. الوسيط(Middleware)

: `auth.middleware.js` * للتحقق من صحة المستخدم المسجل دخوله.

*د. المكتبات المساعدة(Lib)

: `stream.js` * تكامل مع خدمة Stream للدردشة.

: `db.js` * اتصال بقاعدة البيانات.MongoDB

*ج. الطرق(Routes)

: `chat.route.js` * نقاط النهاية (Endpoints) للدردشة.

: `user.route.js` * نقاط النهاية لإدارة المستخدمين والصداقات.

: `auth.route.js` * نقاط النهاية للمصادقة.

*ب. المتحكمات(Controllers)

: `chat.controller.js` * يعالج عمليات الدردشة.

: `user.controller.js` * يعالج عمليات الصداقات والمستخدمين.

: `auth.controller.js` * يعالج عمليات التسجيل، تسجيل الدخول، الخروج، والتكامل.

*أ. نماذج البيانات(Models)

: `FriendRequest.js` * نموذج لطلبات الصداقة.

: `User.js` * نموذج لمستخدم التطبيق.

.2نماذج البيانات:(Models)

*ب. نموذج طلب الصداقة(FriendRequest)

```javascript
const friendRequestSchema = new mongoose.Schema({
status: { type: String, enum: ["pending", "accepted"], default: "pending" },
recipient: { type: mongoose.Schema.Types.ObjectId, ref: "User", required: true },
sender: { type: mongoose.Schema.Types.ObjectId, ref: "User", required: true }
});
```

*أ. نموذج المستخدم(User)

```javascript
const userSchema = new mongoose.Schema({
friends: [{ type: mongoose.Schema.Types.ObjectId, ref: "User" }],
isOnboarded: { type: Boolean, default: false },
location: { type: String, default: "" },
```

learningLanguage: { type: String, default: "" },

nativeLanguage: { type: String, default: "" },

profilePic: { type: String, default: "" },

bio: { type: String, default: "" },

password: { type: String, required: true, minlength: 6 },

email: { type: String, required: true, unique: true },

fullName: { type: String, required: true }

});

```

الوظائف:

* `matchPassword()`: مقارنة كلمة المرور المدخلة مع المشفرة.

* `pre("save")`: تشفير كلمة المرور قبل حفظها.

---

3.المتحكمات:(Controllers)

*ج. متحكم الدردشة(chat.controller.js)

* `getStreamToken()`: ينشئ رمز وصول لخدمة Stream للدردشة.

*ب. متحكم المستخدم(user.controller.js)

1. `getOutgoingFriendReqs()`: يعرض طلبات الصداقة الصادرة.

2. `getFriendRequests()`: يعرض طلبات الصداقة الواردة والمقبولة.

3. `acceptFriendRequest()`: يقبل طلب صداقة.

4. `sendFriendRequest()`: يرسل طلب صداقة.

5. `getMyFriends()`: يعرض أصدقاء المستخدم الحالي.
```

: `getRecommendedUsers()` .6 يعرض مستخدمين مقترحين للصداقة.

*أ. متحكم المصادقة(auth.controller.js)

1. الخروج:(logout)

* يمسح كوكي الجلسة.

2. التكامل:(onboard)

* يحدد `isOnboarded` إلى. `true`

* يكمل معلومات المستخدم الأساسية.

3. تسجيل الدخول:(login)

* ينشئ JWT للجلسة.

* يتحقق من البريد الإلكتروني وكلمة المرور.

4. التسجيل:(signup)

* ينشئ JWT (JSON Web Token) للجلسة.

* ينشئ مستخدم في خدمة Stream للدردشة.

* ينشئ حساب مستخدم جديد.

.4نقاط النهاية:(Endpoints)

*ج. مسار الدردشة(chat.route.js)

: `GET /api/chat/token` * الحصول على رمز الدردشة.

*ب. مسار المستخدم(user.route.js)

: `GET /api/users/outgoing-friend-requests` * الحصول على الطلبات الصادرة.

: `GET /api/users/friend-requests` * الحصول على طلبات الصداقة.

: `PUT /api/users/friend-request/:id/accept` * قبول طلب صداقة.

: `POST /api/users/friend-request/:id` * إرسال طلب صداقة.

: `GET /api/users/friends` * الحصول على قائمة الأصدقاء.

: `GET /api/users/` * الحصول على مستخدمين مقترحين.

*أ. مسار المصادقة(auth.route.js)

: `GET /api/auth/me` * الحصول على بيانات المستخدم الحالي.

: `POST /api/auth/onboarding` * تكملة بيانات المستخدم.

: `POST /api/auth/logout` * تسجيل خروج.

: `POST /api/auth/login` * تسجيل دخول.

: `POST /api/auth/signup` * تسجيل مستخدم جديد.

.5الأسئلة الشائعة حول الكود:

*د. أسئلة عامة:

1. كيف يتم التعامل مع الملفات الثابتة في الإنتاج؟

* في `server.js` يتم استخدام `express.static` لخدمة ملفات React المبنية.

2. ما هي ميزات الأمان المستخدمة؟

* تحقق من الصلاحيات قبل تنفيذ العمليات الحساسة.

* استخدام JWT في كوكي آمن.(httpOnly, secure, sameSite)

* تشفير كلمات المرور باستخدام.bcrypt

3. كيف يتم التعامل مع الأخطاء؟

* يتم إرجاع أكواد حالة HTTP مناسبة (مثل 400 للبيانات غير الصالحة، 401 للغير مصرح به).

* يتم استخدام `try/catch` في كل دالة.

4. ما هي أنواع الطلبات (HTTP Methods) المستخدمة؟

* `DELETE`: لم يتم استخدامه هنا ولكن يستخدم عادةً للحذف.

* `PUT`: لتحديث البيانات.

* `POST`: لإنشاء بيانات جديدة.

* `GET`: لاسترجاع البيانات.

*ج. أسئلة حول الدردشة:

1. كيف يتم تكامل الدردشة معStream ؟

* يتم إنشاء رمز وصول عبر. `GET /api/chat/token`

* يتم إنشاء مستخدم في Stream عند التسجيل والتكامل.

*ب. أسئلة حول إدارة الصداقات:

1. كيف يتم عرض قائمة الأصدقاء؟

* عبر. `GET /api/users/friends`

2. كيف يتم قبول طلب صداقة؟

* عبر `PUT /api/users/friend-request/:id/accept` حيث `id:` هو معرّف الطلب.

3. كيف يتم إرسال طلب صداقة؟

* عبر `POST /api/users/friend-request/:id` حيث `id:` هو معرّف المستخدم المستقبل.

*أ. أسئلة حول المصادقة:(Authentication)

1. ما هو الغرض من`protectRoute` middleware ؟

* يحمي نقاط النهاية من الوصول غير المصرح به.

* للتحقق من صحة JWT في كل طلب يحتاج إلى مصادقة.

2. كيف يعمل نظام تسجيل الدخول؟

* يتم إنشاء JWT وتخزينه في كوكي آمن.

* عبر `POST /api/auth/login` مع `email` و.`password`

3. كيف يتم تسجيل مستخدم جديد؟

* يتم التحقق من صحة البريد الإلكتروني وقوة كلمة المرور.
* يتم عبر `POST /api/auth/signup` مع إرسال `email`, `password`, `fullName`.

6.مصطلحات تقنية:

HTTP Status Codes: *أكواد تعبر عن نتيجة الطلب (200 نجاح، 404 غير موجود، إلخ).

CORS (Cross-Origin Resource Sharing): *آلية تسمح بطلبات عبر النطاقات المختلفة.

Stream Chat: *خدمة دردشة جاهزة للتكامل مع التطبيقات.

Mongoose: *مكتبة لتعامل مع MongoDB في Node.js.

JWT (JSON Web Token): *معيار لإنشاء رموز وصول يمكن التحقق منها.

Middleware: *برمجية وسيطة تعالج الطلب قبل وصوله للتحكم النهائي) مثل.(`protectRoute`

Endpoint: *عنوان URL الذي يستقبل طلبات) HTTP مثل.(`/api/auth/signup`

7.تحسينات مقترحة:

1.إضافة نظام تسجيل الأحداث (Logging) أكثر تطوراً.

2.تحسين معالجة الأخطاء بإرجاع رسائل أكثر وصفية.

3.إضافة حدود لمعدل الطلبات (Rate Limiting) للحماية من الهجمات.

4.إضافة نظام إشعارات لطلبات الصداقة.

5.إضافة التحقق من الصلاحيات (Authorization) بجانب المصادقة.(Authentication)

الخلاصة:

يمثل هذا الكود نظامًا متكاملاً لإدارة المستخدمين والصداقات والدردشة مع تطبيق أفضل ممارسات الأمان وتنظيم الكود.