как можно матчить 50 параметров (условно) критериев клиента с данными (характеристиками) от баз данных домов? я просто думаю ценность выбора дома престарелых не тільки у відкритій інфі а у них у кожного клиента якісь кастомні потреби. типу інсульт і треба перевертати кожну годину чи можуть там робити так

Отличный вопрос! Именно кастомные потребности создают основную ценность продукта. Обычные directory сайты показывают только базовую информацию, а семьи ищут специфические услуги. Вот как построить систему матчинга 50+ параметров:

Структура Многоуровневого Матчинга

Уровень 1: Базовые Параметры из CQC API

```
interface BasicCareHomeData {
  // Из CQC API (доступно сразу)
  cqc_ratings: CQCRatings
  care_types: string[] // nursing, residential, dementia
  bed_capacity: number
  location: Coordinates
  specialisms: string[] // learning disabilities, mental health
}
```

Уровень 2: Детальные Возможности (Собираем самостоятельно)

```
interface DetailedCareCapabilities {
  // Medical care capabilities
  stroke_rehabilitation: boolean
  post_surgery_care: boolean
  palliative_care: boolean
  diabetes_management: boolean
  catheter_care: boolean
  peg_feeding: boolean
  oxygen_therapy: boolean
  wound_care_specialist: boolean
  physiotherapy_onsite: boolean
```

```
  speech_therapy: boolean

  // Dementia-specific
  dementia_wandering_management: boolean
  behavioral_support: boolean
  activities_for_dementia: boolean
  secure_garden_access: boolean

  // Physical disabilities
  hoisting_equipment: boolean
  wheelchair_accessibility_all_areas: boolean
  adapted_bathrooms: boolean
  pressure_relief_mattresses: boolean
  turning_schedule_hourly: boolean // Ваш пример!

  // Dietary & lifestyle
  kosher_meals: boolean
  halal_meals: boolean
  vegetarian_options: boolean
  diabetic_meal_plans: boolean
  cultural_activities: string[]
  language_support: string[]

  // Family & social
  overnight_stays_allowed: boolean
  pet_visiting_policy: boolean
  flexible_visiting_hours: boolean
  family_room_available: boolean
  video_calling_support: boolean

  // End-of-life care
  single_rooms_available: boolean
  family_accommodation: boolean
  chaplaincy_services: boolean
  bereavement_support: boolean
}
```

🤖 AI-Powered Data Collection Strategy

Метод 1: Automated Phone/Email Surveys

```
// AI-generated survey система
class CareHomeDetailedSurvey {
  async collectDetailedCapabilities(careHome: CareHome) {
    const questions = await this.generateCustomQuestions(careHome)

    // Отправляем structured questionnaire
    const survey = {
      care_home_id: careHome.id,
      questions: [
        {
          id: "stroke_care",
          question: "Do you provide specialized stroke rehabilitation including regular t
          type: "boolean_with_details",
```

```
          follow_up: "How frequently can residents be turned? (hourly/2-hourly/as-needed)
        },
        {
          id: "cultural_support",
          question: "What cultural and religious accommodations do you provide?",
          type: "multiple_choice",
          options: ["Kosher meals", "Halal meals", "Prayer rooms", "Cultural activities"]
        }
        // ... 50+ questions
      ]
    }

    await this.sendSurveyEmail(careHome.email, survey)
    await this.schedulePhoneFollowup(careHome.phone, survey)
  }
}
```

Метод 2: Web Scraping + AI Analysis

```
// AI анализирует сайты care homes
class CareHomeCapabilityExtractor {
  async extractCapabilitiesFromWebsite(careHome: CareHome) {
    if (!careHome.website) return null

    const content = await this.scrapeCareHomeWebsite(careHome.website)

    // AI prompt для анализа
    const analysis = await openai.chat.completions.create({
      model: "gpt-4o",
      messages: [{
        role: "user",
        content: `Analyze this care home website content and extract specific capabilitie

Website: ${careHome.name}
Content: ${content}

Extract these capabilities (answer true/false/unknown for each):
1. Stroke rehabilitation with hourly turning
2. PEG feeding support
3. Oxygen therapy available 24/7
4. Kosher/Halal meal options
5. Wheelchair accessibility in all areas
6. Dementia wandering prevention systems
7. Family overnight accommodation
8. Pet visiting policies
9. End-of-life care with single rooms
10. Cultural activities and language support

Format as JSON with evidence quotes.`
      }]
    })

    return JSON.parse(analysis.choices[0].message.content)
```

```
  }
}
```

Метод 3: Family/Resident Reviews Mining

```
// Парсим отзывы для извлечения specific capabilities
class ReviewCapabilityExtractor {
  async analyzeReviewsForCapabilities(careHome: CareHome) {
    const reviews = await this.collectReviews([
      'carehome.co.uk',
      'google.com',
      'checkmycare.org.uk'
    ], careHome)

    const capabilities = await openai.chat.completions.create({
      model: "gpt-4o",
      messages: [{
        role: "user",
        content: `Analyze these care home reviews to extract specific care capabilities m

Reviews: ${reviews.join('\n\n')}

Look for mentions of:
- Specific medical care (turning schedules, wound care, medication management)
- Cultural accommodations (food, religious practices, language)
- Accessibility features (equipment, adaptations)
- Family support (visiting, communication, accommodation)
- Staff expertise (specialists, training, responsiveness)

Return as JSON with confidence scores and review quotes as evidence.`
      }]
    })

    return JSON.parse(capabilities.choices[0].message.content)
  }
}
```

Smart Matching Algorithm

Weighted Scoring System

```
interface ClientNeeds {
  // Critical needs (must-have)
  medical_critical: {
    stroke_care_with_turning: { required: boolean, frequency: 'hourly' | '2-hourly' }
    peg_feeding: boolean
    oxygen_24_7: boolean
    diabetes_management: boolean
  }

  // Important preferences (nice-to-have)
  lifestyle_preferences: {
```

```
    cultural_food: string[] // ['kosher', 'halal', 'vegetarian']
    language_support: string[] // ['ukrainian', 'polish', 'hindi']
    pet_visiting: boolean
    flexible_visiting: boolean
  }

  // Comfort factors (bonus points)
  comfort_factors: {
    single_room: boolean
    garden_access: boolean
    activities: string[]
    location_near_family: boolean
  }
}

class SmartCareMatching {
  calculateMatch(careHome: DetailedCareHome, clientNeeds: ClientNeeds): MatchResult {
    let totalScore = 0
    let maxPossibleScore = 0
    const matchDetails: MatchDetail[] = []

    // Critical needs (40% of score)
    const criticalScore = this.scoreCriticalNeeds(careHome, clientNeeds.medical_critical)
    totalScore += criticalScore.score * 0.4
    maxPossibleScore += criticalScore.maxScore * 0.4
    matchDetails.push(...criticalScore.details)

    // Important preferences (35% of score)
    const preferencesScore = this.scorePreferences(careHome, clientNeeds.lifestyle_prefe
    totalScore += preferencesScore.score * 0.35
    maxPossibleScore += preferencesScore.maxScore * 0.35
    matchDetails.push(...preferencesScore.details)

    // Comfort factors (25% of score)
    const comfortScore = this.scoreComfortFactors(careHome, clientNeeds.comfort_factors)
    totalScore += comfortScore.score * 0.25
    maxPossibleScore += comfortScore.maxScore * 0.25
    matchDetails.push(...comfortScore.details)

    const matchPercentage = (totalScore / maxPossibleScore) * 100

    return {
      careHome,
      matchPercentage,
      matchDetails,
      criticalNeedsMet: criticalScore.allMet,
      recommendationLevel: this.getRecommendationLevel(matchPercentage, criticalScore.al
    }
  }

  private scoreCriticalNeeds(careHome: DetailedCareHome, critical: ClientNeeds['medical_c
    const details: MatchDetail[] = []
    let score = 0
    let maxScore = 0
    let allMet = true
```

```
    // Stroke care with specific turning frequency
    if (critical.stroke_care_with_turning.required) {
      maxScore += 10
      if (careHome.stroke_rehabilitation && careHome.turning_schedule_hourly) {
        if (critical.stroke_care_with_turning.frequency === 'hourly') {
          score += 10
          details.push({
            factor: "Hourly turning for stroke care",
            status: "perfect_match",
            evidence: "Care home provides hourly turning schedule as required"
          })
        } else {
          score += 7
          details.push({
            factor: "Stroke care turning frequency",
            status: "partial_match",
            evidence: "Care home provides hourly turning, you requested 2-hourly (exceeds
          })
        }
      } else {
        allMet = false
        details.push({
          factor: "Stroke care with turning",
          status: "not_available",
          evidence: "Care home does not specify turning schedule capability"
        })
      }
    }

    // PEG feeding support
    if (critical.peg_feeding) {
      maxScore += 10
      if (careHome.peg_feeding) {
        score += 10
        details.push({
          factor: "PEG feeding support",
          status: "perfect_match",
          evidence: "Experienced staff for PEG feeding care"
        })
      } else {
        allMet = false
        details.push({
          factor: "PEG feeding support",
          status: "not_available",
          evidence: "PEG feeding capability not confirmed"
        })
      }
    }

    return { score, maxScore, allMet, details }
  }

  private getRecommendationLevel(matchPercentage: number, criticalMet: boolean): string {
    if (!criticalMet) return "not_suitable"
    if (matchPercentage >= 90) return "excellent_match"
    if (matchPercentage >= 75) return "very_good_match"
```

```
    if (matchPercentage >= 60) return "good_match"
    return "basic_match"
  }
}
```

🧠 Client Needs Assessment (AI-Powered)

Smart Questionnaire Flow

```
// AI генерирует personalized questions based на ответы
class PersonalizedNeedsAssessment {
  async conductAssessment(userEmail: string): Promise<ClientNeeds> {
    let currentNeeds: Partial<ClientNeeds> = {}

    // Базовые медицинские вопросы
    const medicalHistory = await this.askMedicalQuestions([
      "Has your loved one had a stroke?",
      "Do they require assistance with feeding?",
      "Are there any cultural or religious dietary requirements?",
      "What mobility aids do they currently use?"
    ])

    // AI анализирует ответы и задает follow-up вопросы
    const followUpQuestions = await this.generateFollowUpQuestions(medicalHistory)

    // Например, если stroke = true:
    if (medicalHistory.stroke) {
      const strokeCare = await this.askStrokeSpecificQuestions([
        "How frequently do they need to be turned to prevent pressure sores?",
        "Do they require physiotherapy?",
        "Are they at risk of falls or wandering?"
      ])

      currentNeeds.medical_critical = {
        ...currentNeeds.medical_critical,
        stroke_care_with_turning: {
          required: true,
          frequency: strokeCare.turning_frequency
        }
      }
    }

    return currentNeeds as ClientNeeds
  }

  private async generateFollowUpQuestions(responses: any): Promise<string[]> {
    const prompt = `Based on these care needs responses, what specific follow-up questior

    Responses: ${JSON.stringify(responses)}

    Generate 3-5 specific questions that will help identify exact care capabilities neede

    const questions = await openai.chat.completions.create({
      model: "gpt-4o",
```

```
      messages: [{ role: "user", content: prompt }]
    })

    return JSON.parse(questions.choices[0].message.content)
  }
}
```

Practical Implementation Strategy

Phase 1: Start with High-Impact Parameters

```
// Focus на наиболее важные differentiators
const HIGH_IMPACT_PARAMETERS = [
  // Medical capabilities (20 parameters)
  'stroke_rehabilitation', 'turning_schedules', 'peg_feeding', 'oxygen_therapy',
  'wound_care', 'diabetes_management', 'catheter_care', 'physiotherapy',

  // Accessibility (10 parameters)
  'wheelchair_access_all_areas', 'hoisting_equipment', 'adapted_bathrooms',
  'hearing_loop_system', 'visual_impairment_support',

  // Cultural & lifestyle (15 parameters)
  'kosher_meals', 'halal_meals', 'cultural_activities', 'language_support',
  'pet_policies', 'visiting_flexibility', 'overnight_family_stays',

  // End-of-life care (5 parameters)
  'single_rooms_available', 'family_accommodation', 'chaplaincy', 'bereavement_support'
]
```

Phase 2: Data Collection Prioritization

1. **Automated collection**: Web scraping + AI analysis (70% coverage)
2. **Direct surveys**: Email questionnaires to care homes (20% coverage)
3. **Phone verification**: For high-value clients (10% coverage)
4. **User feedback**: Crowdsource from families who visited (ongoing)

Phase 3: Confidence Scoring

```
interface ParameterData {
  value: boolean | string | number
  confidence: number // 0-100%
  source: 'website' | 'survey' | 'phone_verified' | 'user_reported'
  last_updated: Date
  evidence?: string
}

// Показываем клиенту confidence level
"✅ Hourly turning schedule: Confirmed (95% confidence)"
```

```
"⚠ Kosher meals: Likely available (70% confidence) - call to confirm"
"? PEG feeding: Unknown - we'll verify for your report"
```

Ценностное Предложение

Вместо: "Here are care homes near you"
Предлагаем: "Here are 3 care homes that can provide hourly turning for stroke patients, kosher meals, and Ukrainian-speaking staff - exactly what your father needs"

Это и есть тот **$100M Offer** который Hormozi описывает - мы решаем очень специфическую проблему ("найти дом престарелых который может справиться с инсультом И культурными потребностями") быстро и точно.

Результат: Семьи готовы платить £299-499 за такую персонализацию, потому что альтернатива - недели звонков по care homes с вопросами "А вы можете переворачивать каждый час?"