



# AnySpi.net 開発ケーススタディ

信頼と実績を可視化する、Laravelベースの占いマッチングプラットフォーム

個人開発 / フルスタック実装 | 使用技術: LARAVEL 11.31 / MYSQL / BOOTSTRAP 5.3

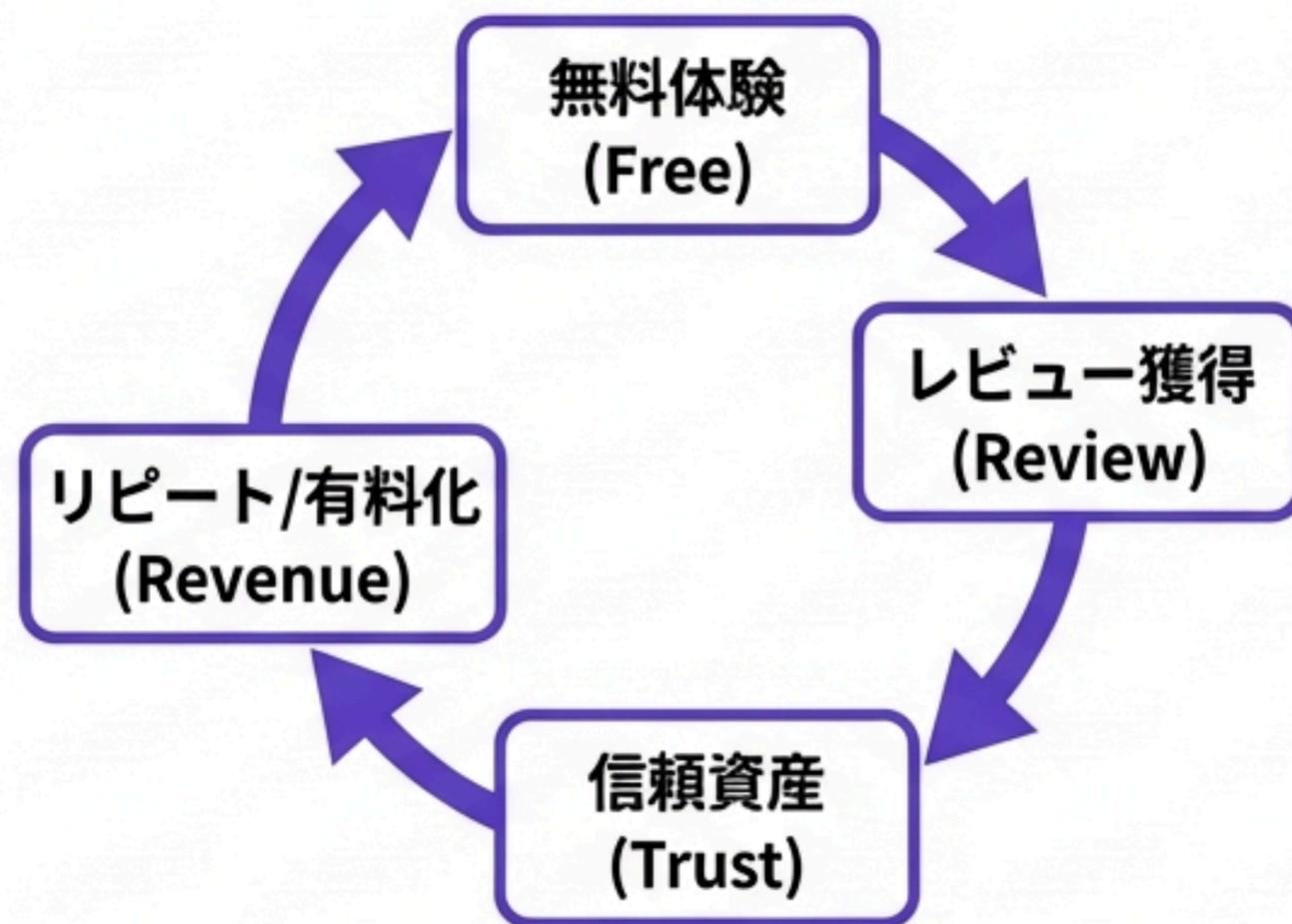
# 市場課題：1兆円市場に潜む「信頼の非対称性」

## THE FRICTION



- **心理的ハードル**：知らない相手への課金に対する不安。
- **コールドスタート問題**：新規占い師は実績（レビュー）がないため、集客が困難。

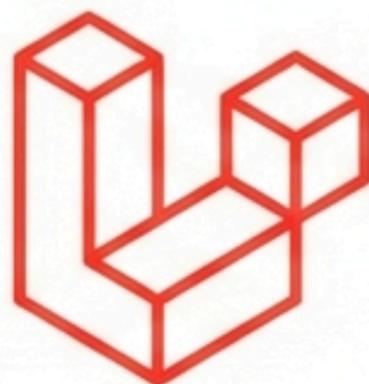
## THE SOLUTION



**AnySpiの解決策**：「無料体験」を起点とした信頼構築モデル。

# 技術スタック：開発スピードと保守性の両立

## Backend



Laravel 11.31  
(Latest Standard)

## Frontend



Bootstrap 5.3 (Admin/User UI)  
+ Tailwind 2.2 (LP)

## Database



MySQL

Architecture Policy: MVPとしての速さを重視し、Viteを除外したシンプルなLaravel UI構成を採用。

複雑性を排除し、ビジネスロジックの実装に集中できる環境を構築。

# データベース設計：拡張性を考慮したスキーマ

1

**Users Table:**  
管理者(Admin)・占い師(Reader)・ユーザー(User)を単一テーブルで管理し、JOINコストを削減。

USERS			
bigint	id	PK	
string	name		
string	email	UK	
boolean	is_reader		占い師フラグ
boolean	is_admin		管理者フラグ
integer	points		現在ポイント
integer	login_streak		連続ログイン数
string	invite	UK	紹介コード

2

**Performance:**  
Eager Loading実装。  
N+1問題を回避。

settings		
bigint	id	PK
string	key	UK
string	value	

partners			
bigint	id	PK	
foreign_id	user_id	FK	
string	name		
date	birth_date		
text	relationship		

login_bonus_logs				
bigint	id	PK		
foreign_id	user_id	FK		
date	login_date	UK		同一日の重複付与禁止

inviters				
bigint	id	PK		
foreign_id	inviter_id	PK		紹介した人
foreign_id	invitee_id	FK		紹介された人
integer	points_awarded			

reviews				
bigint	id	PK		
foreign_id	user_id	FK		相読者
foreign_id	reader_id	FK		占い師(Usersテーブル)
tinyinteger	rating			
text	comment			
boolean	is_public			

3

**Safety:** `user\_id` と `login\_date` の複合ユニーク制約で重複付与を防止。

review_reports				
bigint	id	PK		
foreign_id	review_id	FK		
foreign_id	reporter_id	FK		通報者
string	status			pending/approved/rejected

根拠相手登録

ログイン履歴登録

通報者として記録

紹介者として記録

相読者として記録

クライアントとして記録

占い師として記録

通報対象

# MVP戦略：アジャイルな意思決定とトレードオフ

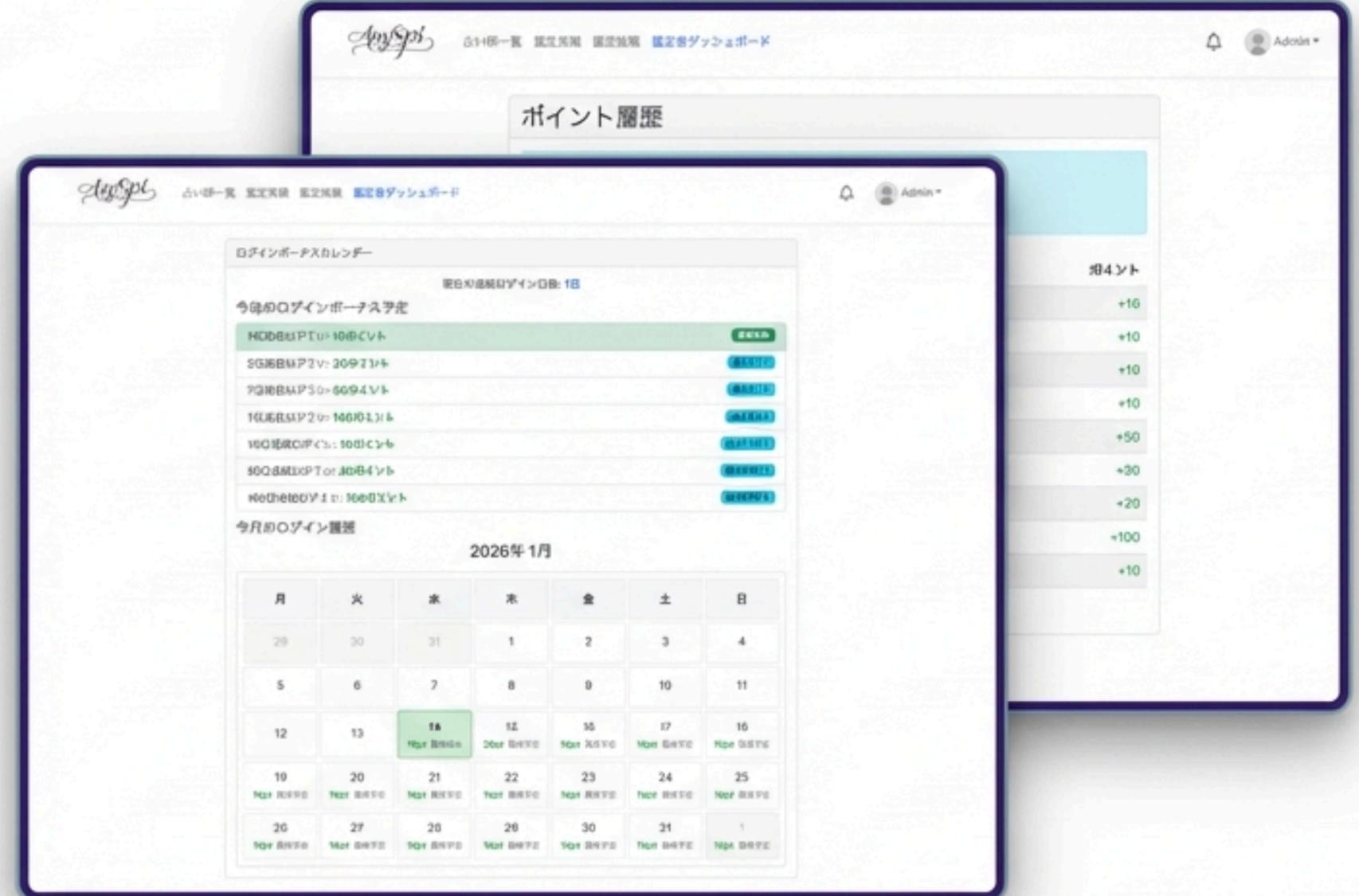
課題	3種類のユーザー（一般/占い師/管理者）の権限管理。
選択	`roles` テーブルによるRBACではなく、`is_admin` / `is_reader` フラグを採用。
理由	スクラム開発の考えに基づき、初期実装コストを最小化。要件が複雑化した時点でRole管理へ移行可能なモジュラーな設計。

```
$table->boolean('is_admin')->default(false);  
$table->boolean('is_reader')->default(false);
```

# ゲーミフィケーション：継続利用を促す仕組み

## Engagement Loop

- **ログインストリーク**: 連続ログイン日数に応じたインセンティブ設計。
- **透明性**: カレンダービューで将来の獲得ポイントを可視化し、翌日のログインモチベーションを向上。
- **実装**: `login_streak` カラムと `last_login_date` による効率的な判定ロジック。



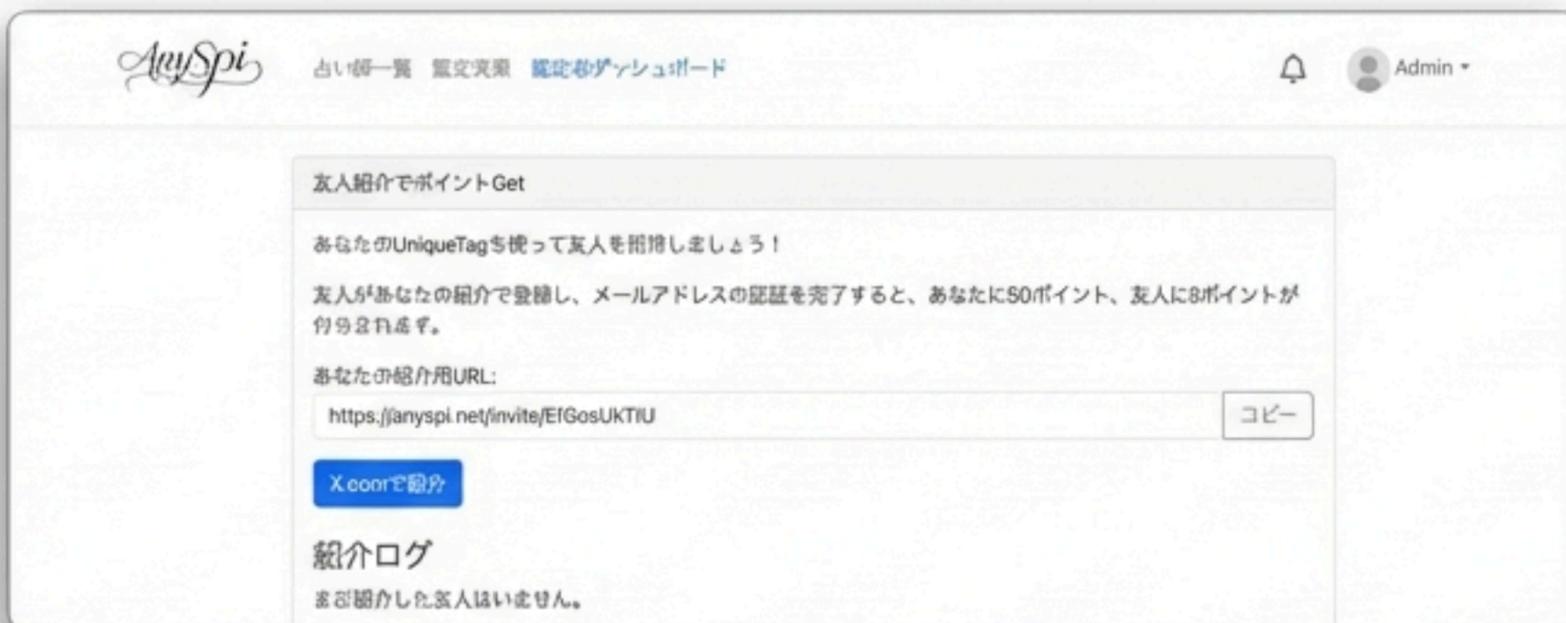
# 信頼エンジン：レビューとポイントの循環システム



## Data Integrity

- `reviews` テーブルは `users` (reader) と `users` (client) を紐付け。
- `review\_reports` テーブルによる通報機能の実装。

# バイラル施策：招待システムとUXへの配慮



- **X連携:** シェアボタンから自動生成されたテキストで即時投稿可能。
- **Cookieトラッキング:** `invite=` パラメータをCookieに保存し、URLをクリーンに保ちつつリダイレクト(`InviteController`の実装)。
- **Code Focus:** `InviteController` 内の二重付与防止ロジックとセッション管理。

```
public function handleInvite(Request $request, $tag)
{
    // UniqueTagから紹介者ユーザーを照会
    $inviter = User::where('invite', $tag)->first();

    if (!$inviter) {
        // 紹介者が見つからない場合はトップページへリダイレクト
        return redirect('/');
    }

    // セッションに紹介者IDを保存
    session(['inviter_id' => $inviter->id]);

    // ログイン済みであれば、直接ポイント付与ロジックへ
    if (Auth::check()) {
        $invitee = Auth::user();
        // 自分自身を招待することはできない
        if ($inviter->id == $invitee->id) {
            session()->forget('inviter_id');
            return redirect('/');
        }

        // 既に招待済みかチェック
        $alreadyInvited = Inviter::where('inviter_id', $inviter->id)
            ->where('invitee_id', $invitee->id)
            ->exists();

        if ($alreadyInvited) {
            session()->forget('inviter_id');
            return redirect('/');
        }

        // ログインしていない場合は登録/ログインページへリダイレクト
        return redirect()->route('register');
    }
}
```

# 鑑定師ダッシュボード：供給側の運用ツール

The image displays two overlapping screenshots of the AnySpi platform. The primary screenshot shows the '鑑定者ダッシュボード' (Reader Dashboard) for a user named 'Admin'. The dashboard includes a profile card with a '鑑定可能' (Available for reading) status, a '統計情報' (Statistics) section showing 0 readings and a 0.0 average rating, and a 'クイックアクション' (Quick Actions) section with buttons for viewing performance, results, and reviews. A secondary screenshot on the right shows the '鑑定者プロフィールを編集' (Edit Reader Profile) page, featuring a profile picture, bio, and various settings like 'プロフィール画像' (Profile Picture), '肩書き' (Title), and '経年数' (Years of Experience).

鑑定者ダッシュボード

Admin  
密碼の永遠王

現在のステータス: 鑑定可能

鑑定を停止する

プロフィールを編集 公開プロフィールを見る

進行中の相談 (0)

現在、進行中の相談はありません。

統計情報

鑑定数	0
平均評価(0)	評価なし
鑑定者登録日	2025/06/15

クイックアクション

- 鑑定実績を見る
- 鑑定結果一覧を見る
- 相談者レビュー一覧
- ブロックリストを管理

鑑定者プロフィールを編集

プロフィール画像

肩書き

相談者へのメッセージ

サイト/SNS URL

プロフィールを更新 キャンセル

ステータス管理：  
「鑑定可能」「停止中」をワンクリックで切り替え。

プロフィール管理：得意な占術（タグ管理）や自己紹介、  
KPI可視化：自身の鑑定数や評価へのクイックアクセス。

# 管理画面：プラットフォームの健全性を可視化



- **KPIモニタリング:** 総ユーザー数、アクティブ鑑定者数、発行済み総ポイント数をリアルタイム表示。
- **UI設計:** Bootstrap 5.3 ベースの直感的な管理インターフェース。

# ユーザーサポート：監査ログを伴うポイント管理

The screenshot displays a user management interface. At the top, there's a 'ユーザー一覧' (User List) header and a 'ダッシュボードに戻る' (Return to Dashboard) button. Below is a table with columns for ID, Name, Email Address, Role, Points, Registration Date, and Actions. A modal window titled 'ユーザーのポイント管理' (User Point Management) is open, showing details for user ID 13. The modal is divided into 'ユーザー情報' (User Information) and 'ポイント編集' (Point Editing). The 'ポイント編集' section includes a '増減ポイント' (Increase/Decrease Points) input field set to 0, a note '正の値: 追加, 負の値: 減算' (Positive value: Add, Negative value: Subtract), and a required '変更理由' (Change Reason) field with a green border and a placeholder example: '例：不正利用による減算、イベント特典による加算' (Example: Deduction due to misuse, Addition due to event benefits). A 'ポイントを更新' (Update Points) button is at the bottom.

ID	名前	メールアドレス	種別	ポイント	登録日	操作
13	[Redacted]	[Redacted]	招待者	110pt	2025/12/22	ポイント詳細 メール送信
12	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]
11	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]
10	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]
3	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]
8	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]
7	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]
6	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]
5	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]	[Redacted]

ユーザー情報	
ID	13
名前	[Redacted]
メールアドレス	[Redacted]
種別	招待者
現在のポイント	110pt

**ポイント編集**

増減ポイント  
0  
正の値: 追加, 負の値: 減算

変更理由  
例：不正利用による減算、イベント特典による加算

ポイントを更新

- **手動操作**：トラブルや返金対応時に管理者がポイントを増減可能。
- **監査証跡**：変更時には「変更理由」の入力を必須化し、不正やミスを防止。履歴はユーザー側のログにも反映。
- **検索機能**：IDやメールアドレスによる高速なユーザー特定。

# マーケティング：セグメント配信と変数埋め込み

お知らせメール送信

送信対象:

件名:

本文:

送信

メール設定

メールフッターテンプレート

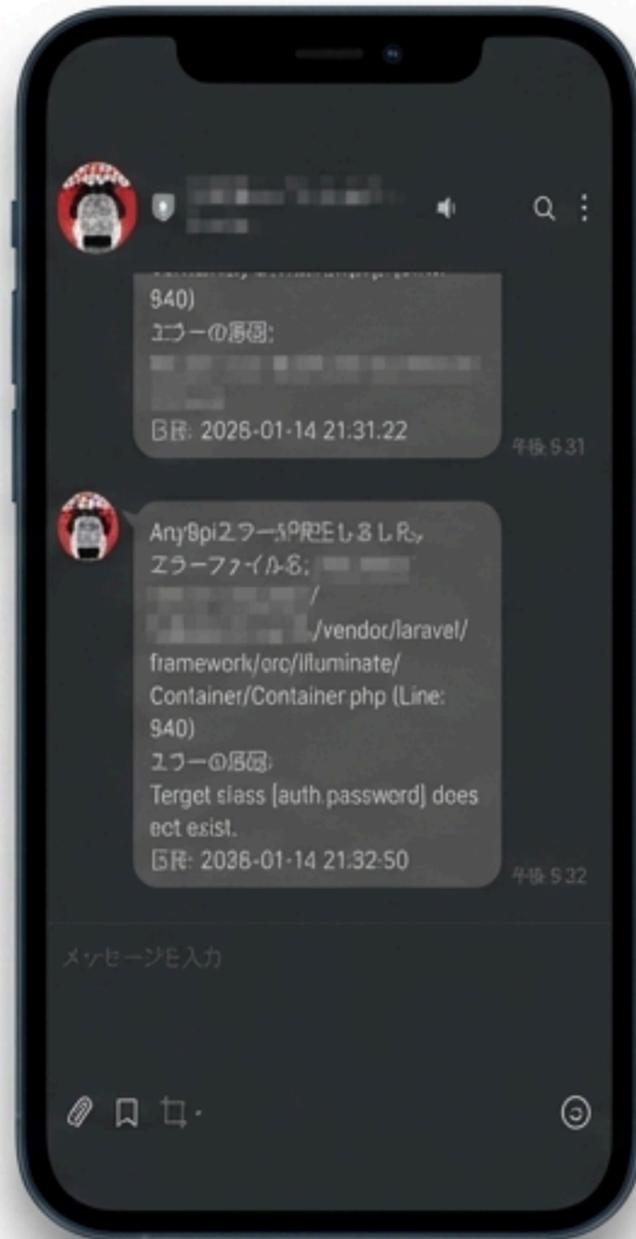
フッターテンプレート

AnySpi - エニスピ  
インターネット情報通信のプラットフォーム  
URL: <https://www.anyspi.net>  
E-MAIL: [info@anyspi.net](mailto:info@anyspi.net)

設定保存

- **セグメント送信**：「一般ユーザー」「鑑定者」「全員」から対象を選択可能。
- **変数差し込み**：{name} や {points} などのタグ機能により、パーソナライズされたメールを送信。
- **共通設定**：管理画面からメールフッターを一元管理し、ブランディングを統一。

# 障害対応：LINE通知による即時検知システム



## Error Handling Flow

1. **Exception Handler:** Laravelの例外処理を拡張し、500エラーをフック。
2. **LINE Messaging API:** エラー詳細（ファイル名、行数、内容）を開発者のLINEへプッシュ通知。
3. **効果:** ユーザーからの問い合わせ前に障害を検知し、修正に着手する「プロアクティブ」な運用体制。

# 今後のロードマップと技術的展望



## Refinement

JetBrains Mono

整合性強化: 招待ロジックにおけるユニーク制約の厳格化 (DBレベル)。



## Security

JetBrains Mono

セキュリティ: メール認証完了後のイベント連動によるポイント付与フローの確立。



## Scalability

JetBrains Mono

スケーラビリティ: ユーザー種別の増加に伴う、Flag管理からRole/Permission管理への移行。



AnySpiは、MVPとしての「スピード」と、将来の拡張に耐えうる「堅牢性」のバランスを重視したプロダクトエンジニアリングの実践です。

# Thank You

ご覧いただきありがとうございます。



杉浦 栄治