種別	評価項目	内容		回答
セキュリティ	クラウド提供業者			
	公的認証取得状況	JIPDECやJQA等で認定している情報処理管理に関する公的認証（ISMS、Pマーク等）を取得	無	現状取得なしだが、1.5年以内にISMS所得予定(現在準備中)
	第三者評価	不正な侵入、操作、データ取得等への対策について、第三者の客観的な評価を得ている	無	なし
	情報取扱い環境	利用者のデータにアクセスできる利用者が適切に限定されていること	有	データベースおよびサーバーのアクセスはAWSでのセキュリティグループにより特定IPアドレス化に制限
	通信の暗号化レベル	システムとやりとりされる通信の暗号化強度	有	アプリケーション及び映像音声データはTLS1.3のみ使用
	ウィルス対策	ウィルススキャン有無（頻度）	有	頻度：AWS Guard Gutyによる常時不正行為の検出
	バックアップデータ	バックアップデータの暗号化、保管場所	有	保管場所：AWS東京リージョン
	利用企業間のデータ分離	企業間の情報隔離	有	データベースによる論理隔離
	当社利用部門（課）			
	利用管理責任者	利用部門に管理責任者の確保	有	代表 中島
	ユーザ管理	1ユーザー1アカウント、退社、移動時の削除管理	有	ユーザー側で物理削除可能
	パスワード	パスワード設定実施管理	有	ユーザー側で変更可能、Emailトークンによる２段階認証も搭載
	監査	管理責任者によるユーザー、パスワード管理定期監査	有	実施サイクル：御社側で対応
サポート体制	クラウド提供業者			
	障害発生時の通知	利用者への連絡方法が決まっている	有	連絡方法：email及びアプリ内のテキストチャットサービス(Intercom)より通知
	緊急連絡先	障害発生時の緊急問い合わせ先の有無	有	テキストチャットサービス(Intercom)もしくはsupport@voice-ping.comより対応 テキストチャットサービス(Intercom)では営業時間中9:00~18:00に通常5分以内に回答
可用性	クラウド提供業者			
	稼働率	サービス利用時間※計画停止を除く		原則24時間365日で、毎週金曜日 21:00頃に数秒程度接続が切れる場合あり
		SLAの基準が設けられている	有	詳細：現状99.9%以上で、ユーザーがサービスを利用できない(音声疎通できる状態でない)状態と定義
		稼働率、ダウンタイム等の実績		現状99.99%以上で、昨年12月のサービス運用以来数十分以上のサーバーダウン等は一度も発生なし
		障害時のリカバリー体制、手順の確立	有	ソースコードの差分バックアップ、DBのバックアップがあり、24時間いつでも即座にロールバック(復旧)可能 また、すべてのサーバープロセスやシステム状態はAWS Cloudwatchもしくは自前のアラートシステムにより全状態が可視化されており、万が一ダウンがあった場合にはアラート通知
		過去３年間で重大障害の発生	無	ありません。
	計画停止	事前通知の日程	無	毎週金曜日 21:00頃をアップデート時間と定めており、数秒程度接続が切れる場合あり その他、営業時間中に万が一アップデートが必要な場合は、Emailもしくはテキストチャットサポートによるリアルタイム通知
データ保護	クラウド提供業者			
	バックアップ	バックアップ実施サイクル、保管期間		実施サイクル：AWS Databaseにより1ヶ月単位のバックアップ 保管期間：サーバーへのアクセスログは永続的にS3に保存
	BCP対策	バックアップデータの遠隔保管	有	AWS Database上でマルチリージョンでクラウド上に遠隔保存

■ ウェブアプリケーションのセキュリティ実装 チェック項目（1/3）

No	脆弱性の種類	対策の性質	チェック	実施項目
1	SQLインジェクション	**根本的解決**	※ ○対応済	○ **SQL文の組み立ては全てプレースホルダで実装する。** ※対応不要の場合の理由(　　) VP対応: バックエンドでミドルウェアのORMでDBの操作を実施。
				○ **SQL文の構成を文字列連結により行う場合は、アプリケーションの変数をSQL文のリテラルとして正しく構成する。** ※対応不要の場合の理由(　　) VP対応: バックエンドでミドルウェアのORMでDBの操作を実施。
		根本的解決	○ 対応済	○ ウェブアプリケーションに渡されるパラメータにSQL文を直接指定しない。 ※対応不要の場合の理由(　　) VP対応: バックエンドでミドルウェアのORMでDBの操作を実施。
		保険的対策	○ 対応済	エラーメッセージをそのままブラウザに表示しない。 VP対応: シングルページアプリケーションを利用しているので、フロントエンドサイドでは直接エラー表示をするような実装は意図的に行わない限り発生しない
		保険的対策	○ 対応済	データベースアカウントに適切な権限を与える。 VP対応:AWS RDBで、READ、WRITEが必要な権限アカウントを作成済み
2	OSコマンド・インジェクション	**根本的解決**	○ 対応済	○ **シェルを起動できる言語機能の利用を避ける。** ※対応不要の場合の理由(　　) VP対応: シェルの起動は基本的に行う必要がないので、存在しない。シェルに近いレイヤーのロジックが必要な場合は安全性が確認されたライブラリを通してのみ実行。
		保険的対策	○ 対応済	○ シェルを起動できる言語機能を利用する場合は、その引数を構成する全ての変数に対してチェックを行い、あらかじめ許可した処理のみを実行する。 VP対応: シェルに近いレイヤーのロジックが必要な場合は安全性が確認されたライブラリを通してのみ実行。
3	パス名パラメータの未チェック／ディレクトリ・トラバーサル	**根本的解決**	※ ○ 対応済	○ **外部からのパラメータでウェブサーバ内のファイル名を直接指定する実装を避ける。** **VP対応: 外部のホスティングもしくはDBにデータはあり、WEBサーバー内ファイルに直接アクセスする機能などは存在しない。**
				○ **ファイルを開く際は、固定のディレクトリを指定し、かつファイル名にディレクトリ名が含まれないようにする。** ※対応不要の場合の理由(　　) VP対応: 外部のホスティングもしくはDBにデータはあり、WEBサーバー内ファイルに直接アクセスする機能などは存在しない。
		保険的対策	○ 対応済	○ ウェブサーバ内のファイルへのアクセス権限の設定を正しく管理する。 VP対応: 外部のホスティングもしくはDBにデータはあり、WEBサーバー内ファイルに直接アクセスする機能などは存在しない。また、バックエンドプロセスがアクセスできるファイル権限も指定されている。
		保険的対策	○ 対応済	○ ファイル名のチェックを行う。 VP対応: ファイル名はランダム名を入れ、予測不可能な形式になっている。
4	セッション管理の不備	**根本的解決**	○ 対応済	○ **セッションIDを推測が困難なものにする。** **VP対応: 予測不可能で、十分な長さのランダム文字列を利用**
		根本的解決	○ 対応済	○ **セッションIDをURLパラメータに格納しない。** **VP対応: Cookieを使用**
		根本的解決	○ 対応済	○ **HTTPS通信で利用するCookieにはsecure属性を加える。** **VP対応: Secure属性を使用**
		根本的解決	※ ○ 対応済	○ **ログイン成功後に、新しくセッションを開始する。** **VP対応: 非ログインアクセス可能ページと、そうでないページは分離済み**
				○ **ログイン成功後に、既存のセッションIDとは別に秘密情報を発行し、ページの遷移ごとにその値を確認する。** **VP対応: 非ログインアクセス可能ページと、そうでないページは分離済み**
		保険的対策	○ 対応済	○ セッションIDを固定値にしない。 VP対応: ランダム文字列で1年期限
		保険的対策	○ 対応済	○ セッションIDをCookieにセットする場合、有効期限の設定に注意する。 VP対応: ランダム文字列で1年期限

※ このチェック項目の「対応済」のチェックは、実施項目のいずれかを実施した場合にチェックします。

■ ウェブアプリケーションのセキュリティ実装 チェック項目（2/3）

No	脆弱性の種類		対策の性質	チェック	実施項目
5	クロスサイト・スクリプティング	HTMLテキストの入力を許可しない場合の対策	**根本的解決**	○ 対応済	○ **ウェブページに出力する全ての要素に対して、エスケープ処理を施す。** **VP対応: 対応するフロントエンドライブラリを使用**
			根本的解決	○ 対応済	○ **URLを出力するときは、「http://」や「https://」で始まるURLのみを許可する。** **VP対応: AWS ELBでhttps強制**
			根本的解決	○ 対応済	○ **<script>…</script> 要素の内容を動的に生成しない。** **VP対応: 対応するフロントエンドライブラリを使用**
			根本的解決	○ 対応済	○ **スタイルシートを任意のサイトから取り込めるようにしない。** **VP対応: 対応するフロントエンドライブラリを使用**
			保険的対策	○ 対応済	○ 入力値の内容チェックを行う。 VP対応: 対応するフロントエンドライブラリを使用
		HTMLテキストの入力を許可する場合の対策	**根本的解決**	○ 対応済	○ **入力されたHTMLテキストから構文解析木を作成し、スクリプトを含まない必要な要素のみを抽出する。** ※対応不要の場合の理由(　　) **VP対応: 対応するフロントエンドライブラリを使用**
			保険的対策	○ 対応済	○ 入力されたHTMLテキストから、スクリプトに該当する文字列を排除する。 VP対応: 対応するフロントエンドライブラリを使用
		全てのウェブアプリケーションに共通の対策	**根本的解決**	○ 対応済	○ **HTTPレスポンスヘッダのContent-Typeフィールドに文字コード（charset）の指定を行う。** ※対応不要の場合の理由(　　) VP対応: バックエンド側で指定
			保険的対策	○ 対応済	○ Cookie情報の漏えい対策として、発行するCookieにHttpOnly属性を加え、TRACEメソッドを無効化する。 VP対応: バックエンド側で指定
			保険的対策	○ 対応済	○ クロスサイト・スクリプティングの潜在的な脆弱性対策として有効なブラウザの機能を有効にするレスポンスヘッダを返す。 VP対応: バックエンド側で指定
6	CSRF （クロスサイト・リクエスト・フォージェリ）		**根本的解決**	※ ○ 対応済	○ **処理を実行するページを POST メソッドでアクセスするようにし、その「hidden パラメータ」に秘密情報が挿入されるよう、前のページを自動生成して、実行ページではその値が正しい場合のみ処理を実行する。** **VP対応: バックエンド側で指定**
					○ **処理を実行する直前のページで再度パスワードの入力を求め、実行ページでは、再度入力されたパスワードが正しい場合のみ処理を実行する。** **VP対応: バックエンド側で指定**
					○ **Refererが正しいリンク元かを確認し、正しい場合のみ処理を実行する。** **VP対応: バックエンド側で指定**
			保険的対策	○ 対応済	○ 重要な操作を行った際に、その旨を登録済みのメールアドレスに自動送信する。 VP対応: Emailアドレス変更などは指定のEmailアドレスに通知
7	HTTPヘッダ・インジェクション		**根本的解決**	※ ○ 対応済	○ **ヘッダの出力を直接行わず、ウェブアプリケーションの実行環境や言語に用意されているヘッダ出力用APIを使用する。** **VP対応: バックエンドミドルウェアで指定**
					○ **改行コードを適切に処理するヘッダ出力用APIを利用できない場合は、改行を許可しないよう、開発者自身で適切な処理を実装する。** **VP対応: バックエンドミドルウェアで指定**
			保険的対策	○ 未対応	外部からの入力の全てについて、改行コードを削除する。

※ このチェック項目の「対応済」のチェックは、実施項目のいずれかを実施した場合にチェックします。

■ ウェブアプリケーションのセキュリティ実装 チェック項目（3/3）

No	脆弱性の種類	対策の性質	チェック	実施項目
8	メールヘッダ・インジェクション	**根本的解決**	※ ○ 対応済	○ **メールヘッダを固定値にして、外部からの入力はすべてメール本文に出力する。** **VP対応: G Suite GmailによるEmail送信をバックエンドより行い、ユーザー側で任意のEmail送信は現状できない**
				○ **ウェブアプリケーションの実行環境や言語に用意されているメール送信用APIを使用する（8-(i) を採用できない場合）。** **VP対応: G Suite GmailによるEmail送信をバックエンドより行い、ユーザー側で任意のEmail送信は現状できない**
		根本的解決	○ 対応済	○ **HTMLで宛先を指定しない。** **VP対応: G Suite GmailによるEmail送信をバックエンドより行い、ユーザー側で任意のEmail送信は現状できない**
		保険的対策	○ 未対策	外部からの入力の全てについて、改行コードを削除する。
9	クリックジャッキング	**根本的解決**	※ ○ 対応済み	**HTTPレスポンスヘッダに、X-Frame-Optionsヘッダフィールドを出力し、他ドメインのサイトからのframe要素やiframe要素による読み込みを制限する。** **VP説明: Iframe等が将来的にVP機能として追加搭載される可能性はあり**
				○ **処理を実行する直前のページで再度パスワードの入力を求め、実行ページでは、再度入力されたパスワードが正しい場合のみ処理を実行する。** **VP説明: 有効なセッションがないと、実行がそもそも不可**
		保険的対策	○ 対応済	○ 重要な処理は、一連の操作をマウスのみで実行できないようにする。 VP説明: ワークスペースやユーザー削除の場合はキーボードタイプでDELETEなどが必要
10	バッファオーバーフロー	**根本的解決**	※ ○ 対応済	○ **直接メモリにアクセスできない言語で記述する。** **VP説明: Node.jsを使用しており、直接メモリにアクセスできない**
				直接メモリにアクセスできる言語で記述する部分を最小限にする。 ※対応不要の場合の理由(　　) VP説明: Node.jsを使用しており、直接メモリにアクセスできない
		根本的解決	○ 対応済	○ **脆弱性が修正されたバージョンのライブラリを使用する。** ※対応不要の場合の理由(　　) **VP説明: Githubの定期ライブラリチェックにより、PRが脆弱性ライブラリに対して挙げられるので適宜ライブラリの更新を実施**
11	アクセス制御や認可制御の欠落	**根本的解決**	○ 対応済	○ **アクセス制御機能による防御措置が必要とされるウェブサイトには、パスワード等の秘密情報の入力を必要とする認証機能を設ける。** ※対応不要の場合の理由(　　) VP説明: マネージャー権限に対しては2段階Emailトークン認証の利用が可能
		根本的解決	○ 対応済	○ **認証機能に加えて認可制御の処理を実装し、ログイン中の利用者が他人になりすましてアクセスできないようにする。** ※対応不要の場合の理由(　　) **VP説明: マネージャー権限に対しては2段階Emailトークン認証の利用が可能**

※ このチェック項目の「対応済」のチェックは、実施項目のいずれかを実施した場合にチェックします。