WebPerformer-NX
開発の流れ解説資料
～プロトタイプ開発のススメ～

2025年10月31日

キヤノンITソリューションズ株式会社

アジェンダ

本資料について

本資料はWebPerformer-NXにおけるシステム開発の流れの一例を解説する資料です。

「複数名の開発者によるプロトタイプ開発手順」について解説しています。

◆ 本資料の対象読者

- WebPerformer-NX サービスに関する基本的な知識を有していること。
- 一般的なシステム開発経験、もしくは同等の知識を有していること。

◆ 前提

- WebPerformer-NX バージョン 3.2.1時点での情報です。

本資料について

◆参考資料

- WebPerformer-NXサービス仕様書
  (https://support.canon-its.co.jp/product/web_performer_nx/faq/pk1/?サービス利用について)
- WebPerformer-NX利用サービスプラン表
  (https://support.canon-its.co.jp/product/web_performer_nx/faq/pk1/?サービス利用について)
- WebPerformer-NXオンラインマニュアル
  (https://docs.webperformer.jp/ja)
- 【WPNX】複数名で同一アプリケーションを開発、変更管理を行うための参考資料
  (https://support.canon-its.co.jp/product/web_performer_nx/faq/pk1/?技術資料)
- 【WPNX】処理を共通化するための参考資料
  (https://support.canon-its.co.jp/product/web_performer_nx/faq/pk1/?技術資料)

用語の定義について

プロトタイプ開発

クライアントからのシステム要件に対し、開発の早い段階で試作品を作成し、フィードバックを反映させながら、アプリケーションを完成させる開発手法です。
試作品を作成することで、クライアントは具体的なシステムをイメージすることができ、開発者もクライアントから直接フィードバックを受けることで、スムーズに開発を進めることができます。

クライアント（ユーザ）

システム要件を決定し、開発者に伝えます。
作成された成果物が要件を満たしているか評価を行い、フィードバックします。
画面UIをデザインするスキルをお持ちであればプロトタイプを作成して、開発者に提示することもできます。

【WebPerformer-NXであると良いスキル】
・画面UI設計/デザイン

開発者

クライアントの要件を満たすシステムを開発します。
プロトタイプを作成し、ユーザテストのフィードバックを反映することを繰り返し、システムを完成に導きます。

【WebPerformer-NXで必要なスキル】
・JavaScript言語
・SQL言語

前提条件

◆ クライアント（ユーザ）

- システム要件および機能要件を検討し、判断できること。

◆ 開発者

-  に関する基本的な知識を有していること。
- JavaScript言語およびSQL言語の知識を有していること。Webアプリケーションの開発経験があると望ましい。

◆ 開発環境/テスト環境

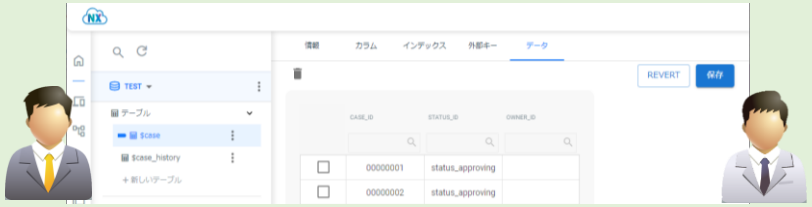
- インターネットブラウザが動作するPCおよびインターネット環境への接続ができること。
- 稼働環境詳細は、WebPerformer-NX オンラインマニュアルをご確認ください。（P.4）

開発局面とWebPerformer-NXの強み

開発の流れ

① システム構想（アイデアを考えて動かせる場面）

- 環境を無償で利用できる
- 簡易的に画面モックを作成できる
- 作成した画面モックを基に複雑度を判断し、工数算出できる

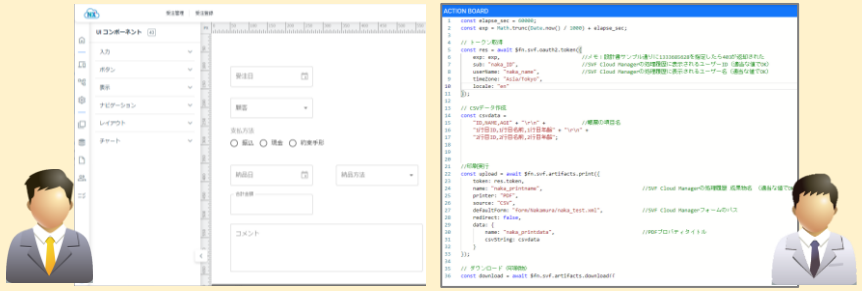


② プロジェクト準備（開発の場面）

- PCからクラウドに繋げばすぐに利用できる
- セキュリティ検査済みのクラウド実行環境
- PC、スマホからの動作を保証

③ 開発期間（スピードの求められる開発の場面）

- 画面とビジネスロジックで作成範囲が分かれており、デザイナーと開発者で分業できる
- 開発、本番環境を運用する上でのリスクを削減できる
  - 負荷分散設計不要（サーバレスサービスで構成）
  - 環境チューニング不要(利用負荷に合わせて選べるサービスプラン)
  - 可用性(環境アラート対応は24時間)
  - セキュリティ、脆弱性への対応（環境側で対応）
- 処理の共通化/生成AIの活用
  - 処理を共通化し、繰り返し同じ処理の記載を削減
  - アクションに必要なコードを生成AIで作成し、開発者はよりクリエイティブな作業に集中



何故プロトタイプ開発手順を勧めるのか

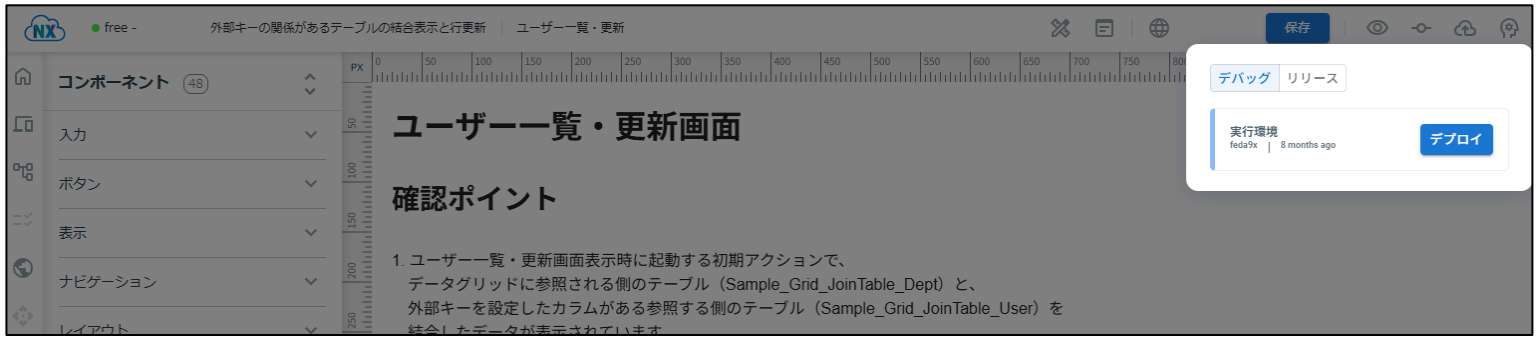
◆実際に動作する試作品の作成が簡単

- 画面の部品をドラッグアンドドロップで配置
- 画面遷移のアクションは、コードを書くことなくプロパティ設定でも可

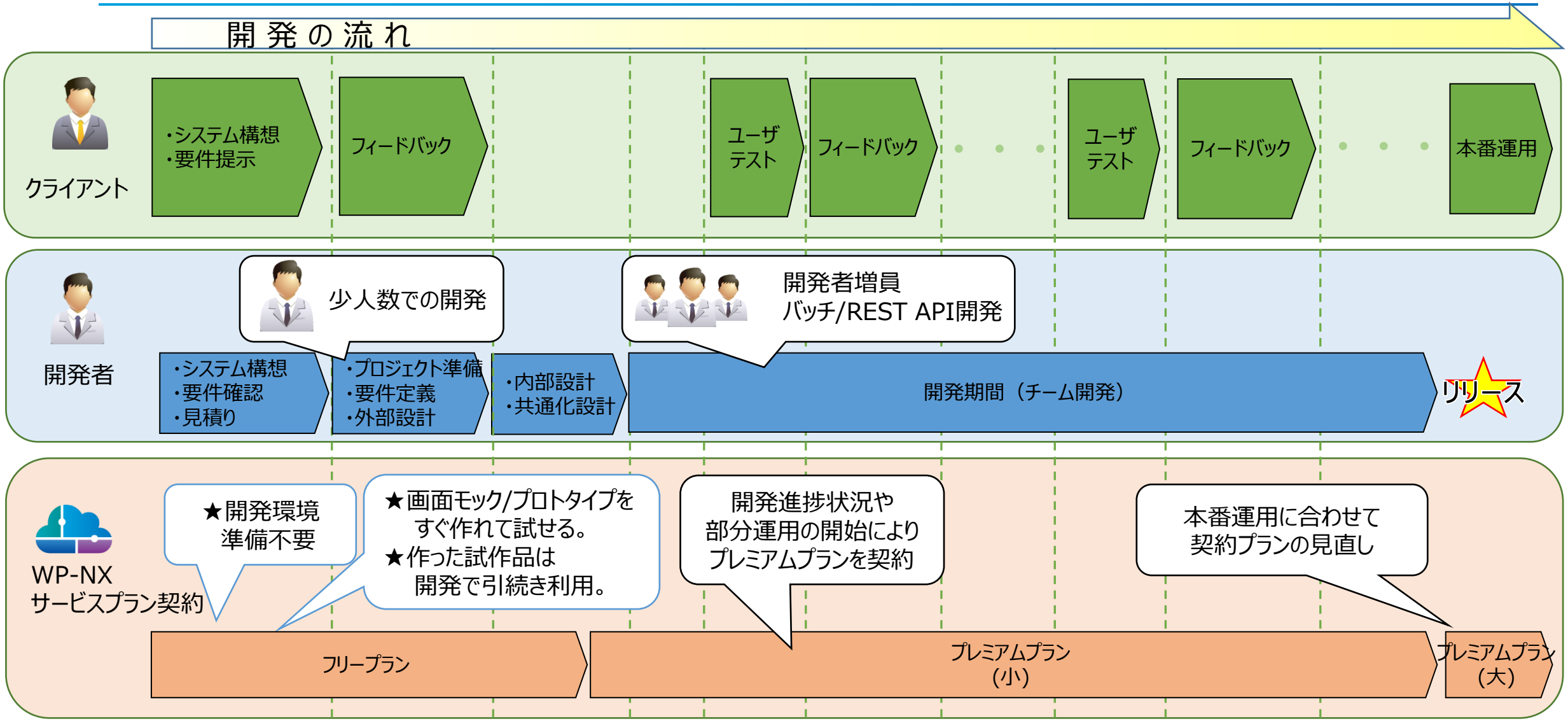
➡ クライアントに共有して説明しながら、その場で簡単に画面UIのデザインを変更することも可能

- ボタン一つでデプロイ

➡ 開発者が簡単にクライアントが確認できる状態にすることができる
クライアントがURLにアクセスし、デザインと操作性を簡単に確認できる



プロトタイプ開発における WebPerformer-NX の適用イメージ



Tips（1）

◆ 見積りについて

WebPerformer-NXで開発を進めるにあたって、作業工数の見積についての考え方をご紹介します。
基本的に、一般的なWebアプリケーション開発と同様に、作成する画面数やデータ構造、機能の複雑さといった点から見積もります。
見積精度を上げるためには、以下の点に注意すると良いでしょう。

- **WebPerformer-NXで実施できることと、できないことを見極める。**
- **機能の複雑さは、作成するアクション数（メインボード、スクリプト、SQL、REST）に落とし込む。**
- **認証UI画面など、NX標準機能の画面を活用できるかどうか検討する。**
- **WebPerformer-NXの標準機能で実現できない要件は、外部ライブラリの活用や RESTアクションを使用した外部連携を検討する。**
- **プロトタイプ開発手法における、レビュー時間、フィードバック時間などを盛り込む。**
- **運用管理機能について、WebPerformer-NXで実施できることを確認し、システム化の要否を判断する。**
- **帳票出力および帳票設計について、SVF Cloudの活用を検討する。**

また、お客様固有のセキュリティ要件に対する適合について、セキュリティチェックリストの記入等でご協力可能ですので、
担当営業までご相談ください。

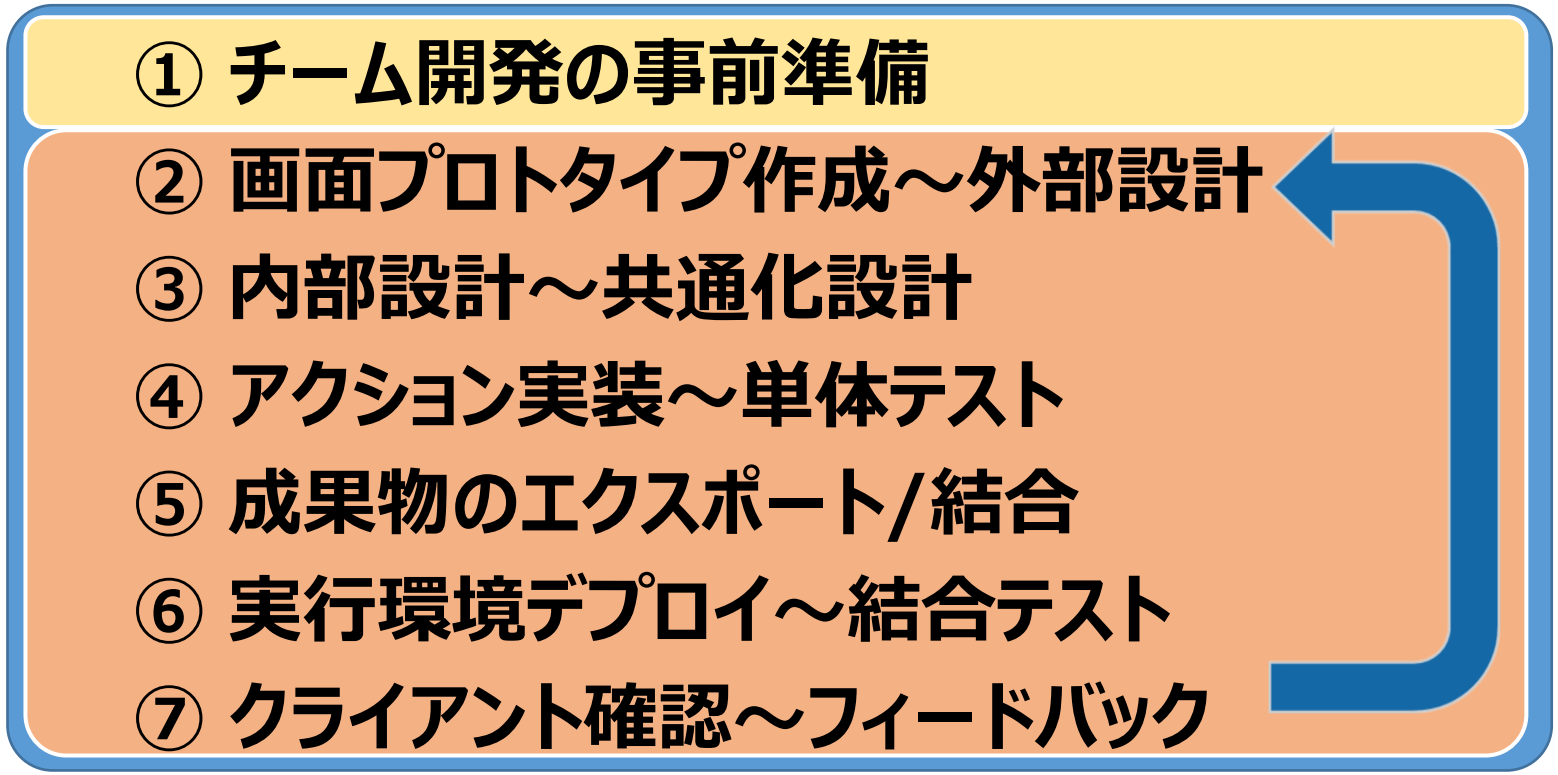
プロジェクト準備

WebPerformer-NXで開発を行う場合でも、通常のシステム開発と同様に、以下のような項目を検討して、プロジェクトの準備を行います。

1. **プロジェクトの目的**
   通常のシステム開発の目的に加えて、ローコード開発ツールを用いることのメリット（短納期、低コスト、高品質な開発の実現）もあげられるでしょう。
2. **システムの概要**
   実現するシステムの機能だけでなく、WebPerformer-NXを用いた運用についても検討します。
3. **開発体制/役割**
   プロトタイプ開発では、クライアントと開発者の役割を明確にします。
   また、チーム開発で作業する場合には、リポジトリの運用ルールについて検討します。
4. **スケジュール**
5. **品質指標**
   基本的には、機能網羅テストの実施を推奨します。指標については、ソースコード行数やステップ数での算出が難しいため、UI画面/バッチジョブ/REST API毎の機能数や機能の複雑さから指標値を策定します。
   指標値の算出については過去実績からの導出が望ましいですが、実績がない場合、開発の一部をサブタスク化して品質サンプルとするなどの方法を検討してください。
   アクション内のJavaScriptコード記述内容については、一般的な手法（コードレビューなど）を適用できると思います。
6. **成果物**
   要件定義書、外部/内部設計書、アプリ定義一式、リソースファイル一式、インポート用DB定義ファイル等・・・。

チーム開発の流れ

チーム体制でプロトタイプ開発を進める場合、以下の流れで進めていきます。



以降の資料で各項目について、説明します。

①チーム開発の事前準備

WebPerformer-NXでチーム開発を行うにあたり、下記の項目を検討/決定します。

1. **開発ガイドラインの策定**
   チーム開発を進めるにあたって、メンバー間の共通認識を高め、必要な規則と手順を明確化するために
   開発ガイドラインを策定します。
2. **開発単位の検討**
   WebPerformer-NXアプリをチームで開発作業するにあたり、メンバー毎の分担単位をどのようにするか、検討します。
   通常のスクラッチ開発のようにソースコードを分散変更管理することはできないため、
   アプリ毎またはUI画面、バッチジョブ、REST API毎で担当を分けることを推奨します。
3. **変更管理方法の検討**
   2.で検討した開発単位を元に、成果物の変更管理方法を検討します。
   アプリ定義はリポジトリ機能を使用して管理することができます。
   また、データベースのテーブル定義や登録データについては、エクスポート機能によりテキストファイルに出力できます。
   変更内容の管理については、外部のタスク/チケット管理ツールなどの導入を検討します。
4. **開発体制の検討/担当者の割り当て**
   開発体制を策定します。
5. **開発環境の準備**
   開発者それぞれで、WebPerformer-NXにサインアップします。
6. **実行環境の準備**
   後述する**「実行環境の共有」機能を利用する場合、プレミアムプラン契約が必要**になります。

Tips（2）

◆ 有償実行環境の共有設定

WebPerformer-NX には、チーム開発者向けに、実行環境の共有機能があります（プレミアムプランのみ）。

- 共有した実行環境上のDBマネージャ、ファイルマネージャ、ユーザマネージャを利用することができます。
- 共有した実行環境上にアプリをデプロイできます。
- 共有した実行環境情報（モニタリング、ログなど）を参照することができます。

有償実行環境の共有設定を行うには2つの方法があります。

① プレミアムプラン申込時に『有償環境を利用するWebPerformer-NXのアカウント情報』項目に追加アカウントを記載する。

② 『すべての環境＞（有償実行環境情報の管理）＞ドメイン,セキュリティと共有＞実行環境の共有先』で、ユーザ追加を行う。

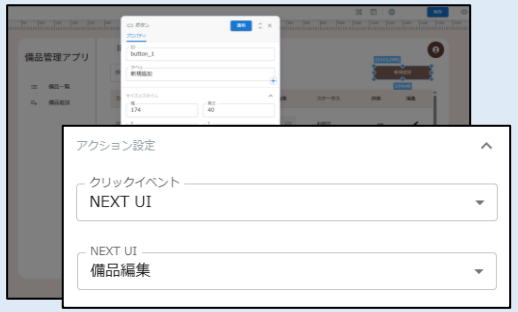
いずれの方法も弊社の運用担当者が共有設定を行います。

②画面プロトタイプ作成～外部設計

開発者それぞれの管理する開発環境で、画面プロトタイプ作成と外部設計を進めます。

1. **アプリを作成（またはインポート）する。**
2. **アプリを共有するためのリポジトリを作成する。**
3. **設計を元に、画面プロトタイプを作成（※）する。**
4. **プロトタイプの作成が完了したら、それぞれの開発者がUIをリポジトリにコミットする。**
5. **リポジトリからアプリをチェックアウトし、実行環境にリリースデプロイする。**
6. **アプリをクライアントに共有し、フィードバックをもらう。**
7. **フィードバックの内容に対して、改善や修正を繰り返す。**
8. **プロトタイプの作成と並行して外部設計を行う。**

※画面プロトタイプの作成



主にクライアントが具体的なシステムをイメージする目的で（プロトタイプを）作成します。
なお、コードを書かずにプロトタイプを作成することが可能です。
下記のような設定を行うのが良いでしょう。

- 必要な項目をドラッグアンドドロップで配置
- 値を表示するコンポーネントのプロパティ設定にて値もしくはデータを設定
- ボタンやリンクのプロパティ設定にて画面遷移を設定

Tips（3）

◆ アプリケーションの共有機能について

WebPerformer-NX には、チーム開発者向けにリポジトリ機能があります。

- アプリケーションを共有することができます。
- アプリケーションのバージョン管理が行えます。

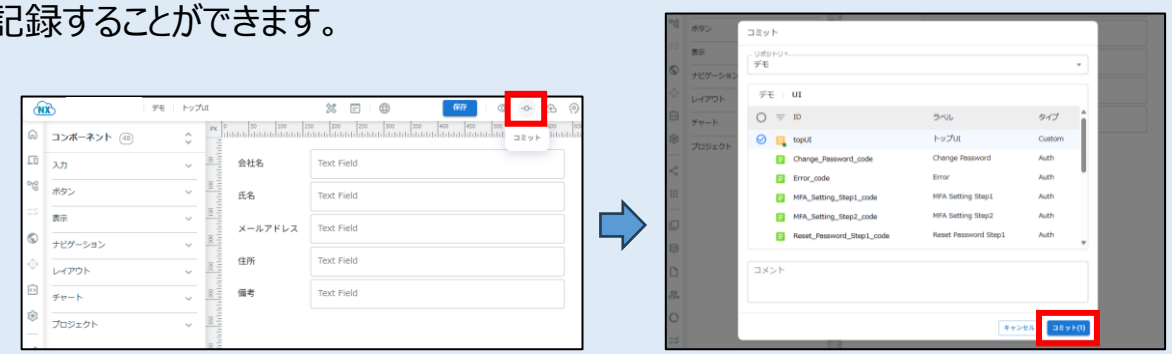
リポジトリ機能内で実施できる主な操作は下記の通りです。

①リポジトリの作成

- 『リポジトリ一覧 ＞ 作成』にてリポジトリを作成します。
- 作成したリポジトリに対してパーミッション設定を行うことで、共有ユーザを管理することができます。

②リポジトリへのコミット

- 『（対象アプリ）＞コミット』から成果物を選択し、
  コミットすることで定義の追加や変更内容をリポジトリに記録することができます。

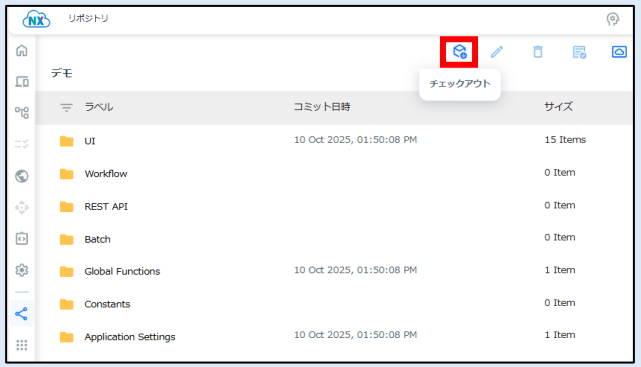


Tips（3）

◆ アプリケーションの共有機能について

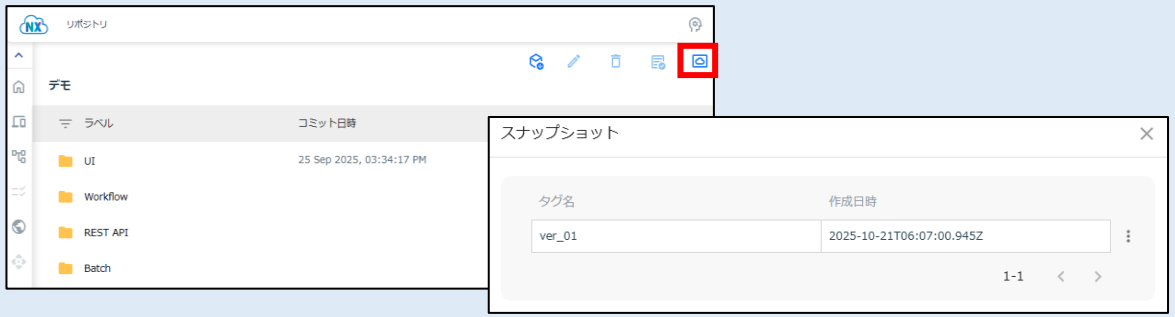
③リポジトリのチェックアウト

- 『リポジトリー覧＞ （共有されたリポジトリ） ＞チェックアウト』からチェックアウトします。
- チェックアウトすることでコミットされた定義を開発環境にコピーすることができます。
- 特定のバージョンを指定し、チェックアウトすることも可能です。



④スナップショット

- 『リポジトリー覧＞ （共有されたリポジトリ） ＞スナップショット』から記録/管理することができます。



※詳細は、【WPNX】複数名で同一アプリケーションを開発、変更管理を行うための参考資料をご参照ください。（P.4）

③内部設計～共通化設計

外部設計で決定した画面や機能をどのように開発し実現するか設計し、
複数の画面共通で使用する部品や処理を洗い出しどのように共通化するか（※）検討します。

※プロトタイプから共通化を検討する方法

1. **分析の実施**
   作成したプロトタイプ・外部設計を元に使用しているコンポーネントや必要な処理を洗い出しします。
   処理の例）データの取得方法、エラーハンドリング、バリデーションルールなど
2. **共通点の特定**
   洗い出したコンポーネントや処理を画面/バッチジョブ/REST API毎に比較し、共通する部分や類似性があるかどうか調査します。
   また、よく使われるパターンや処理の流れを特定し、共通化の候補として検討していきます。
   共通する処理の例）ログ出力処理、メニュー項目の設定処理など
3. **次のステップの計画**
   洗い出した共通部品や処理をどのように実装していくか、検討を進めます。
   定数やグローバル関数等、どのような方法を使って共通化するのかも並行して検討します。
   NXでは共通コンポーネントの機能はありませんが、複数の画面で同じコンポーネントを使用する場合は、
   共通処理用にコンポーネントIDを同じになるように意識します。
   処理の共通化については、【WPNX】処理を共通化するための参考資料（P.4）も併せてご参照ください。

④アクション実装～単体テスト

開発者それぞれの管理する開発環境/実行環境で、開発作業を行います。
作成したプロトタイプのアクション部分を内部設計/共通化設計に従い、実装していきます。

1. **必要に応じてDBを作成し、アプリ設定（※）を変更する。**
2. **アクション部分の開発作業を行う。**
3. **単体テストを行う。**

※規定データベース設定

『アプリケーション設定＞ 一般設定＞ デフォルト設定＞ 規定データベース』を、
実行環境のDB名に合わせて設定します。

各開発者が管理する開発・テスト用の実行環境のDBや、結合テスト用DB、本番用DBなど、フェーズや作業目的に合わせて、DBをご用意ください。

なお、複数のDBを1つの実行環境に作成するには、プレミアムプランが必要になります。
また、規定データベースは、画面のアプリケーションとバッチアプリケーションが設定対象となります。

⑤成果物のエクスポート/結合

開発者それぞれの成果物を結合し、結合テスト環境にデプロイします。

1. **単体テスト完了後にそれぞれの開発者が成果物をリポジトリにコミットする。**
2. **リポジトリからアプリをチェックアウトする**
3. **DBテーブルの修正がある場合は、DBのエクスポートを行う。**
4. **必要に応じて、DBテーブルの定義を結合し、反映する。**
5. **結合テスト用の実行環境に合わせて、アプリ設定（DB設定）を変更する。**

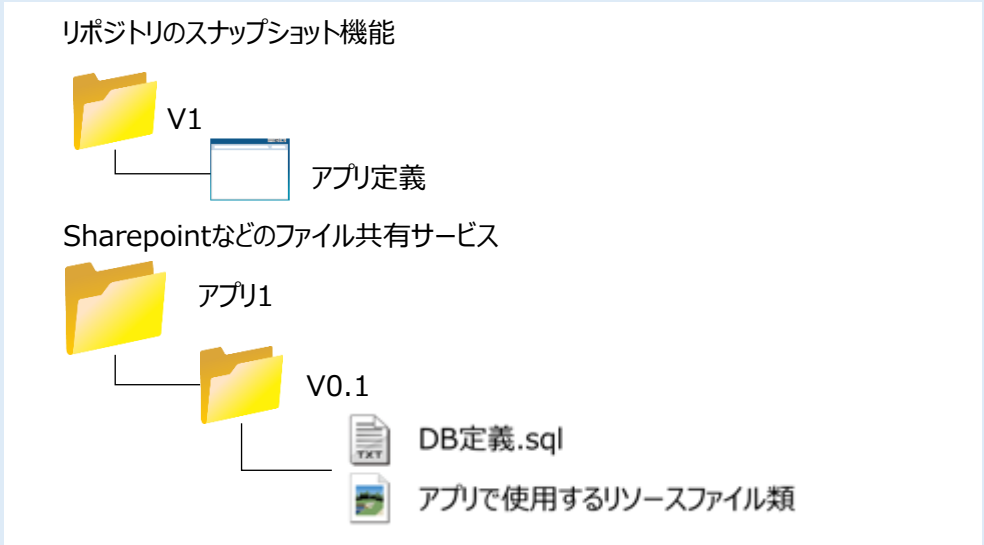
結合方法の詳細は、【WPNX】複数名で同一アプリケーションを開発、変更管理を行うための参考資料をご参照ください。（P.4）

⑥実行環境デプロイ～結合テスト

結合テスト用環境で結合した成果物に対し、結合テストを行います。

1. **結合テスト用の実行環境にリリースデプロイする。**
2. **結合テストを行う。必要に応じて修正し、再デプロイを行う**
3. **テストが完了したら、成果物をエクスポートする（アプリエクスポート機能）。成果物をバージョン管理する。**
   アプリ定義のバージョン管理には、リポジトリのスナップショット機能をご利用いただけます。

◆ **成果物管理イメージ**



Tips（4）

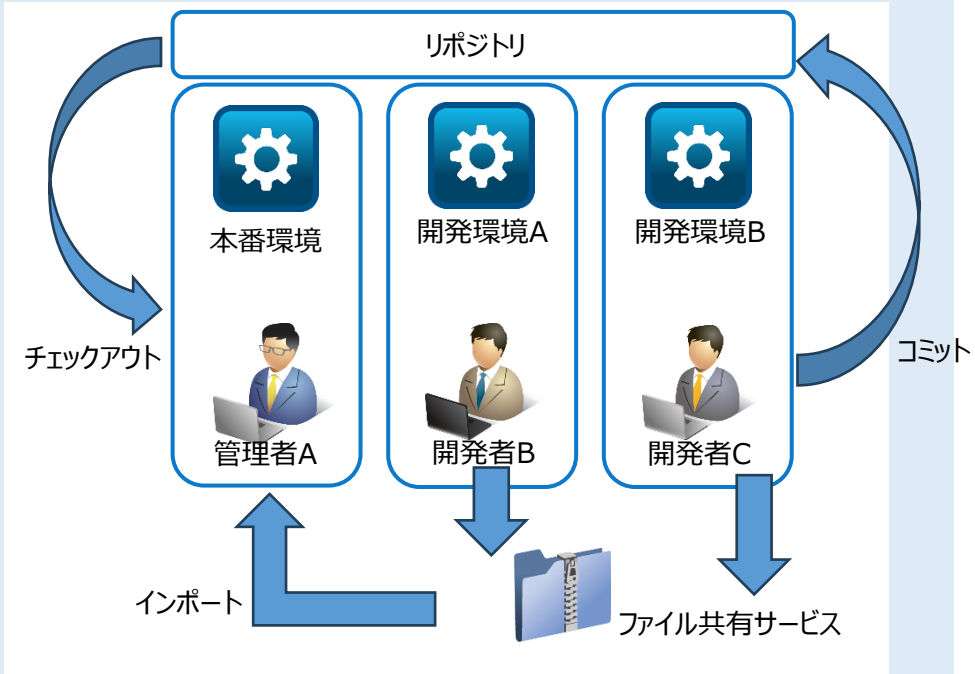
◆ 成果物の変更管理について

WebPerformer-NX 成果物の変更管理を行うにあたり、留意すべきポイントを記載します。

① WebPerformer-NXでは、アプリ定義をリポジトリ機能を用いて管理することができる。

② DB定義およびファイルのバージョン管理、差分管理機能は提供されていないため、
定義をエクスポートして外部サービスで管理する必要がある。

例）エクスポートしたファイルや、結合した定義ファイルを、共有フォルダで管理する。
変更内容は、タスク/チケット管理システムに記録し、ファイル名と紐づける。等



※詳細は、【WPNX】複数名で同一アプリケーションを開発、変更管理を行うための参考資料をご参照ください。（P.4）

⑦クライアント確認～フィードバック

結合テスト環境をクライアントに公開し、作成した成果物の内容をデモを交えて説明します。

クライアントから意見や要望を収集し、リリースするか、もう一度開発作業を行うか、判断します。

開発終了判定～リリース準備

プロジェクト計画で策定した目的と品質目標を満たしているかを判定し、開発を終了できるかを判断します。

（ご参考：WebPerformer-NXにおける責任共有モデルの考え方※）

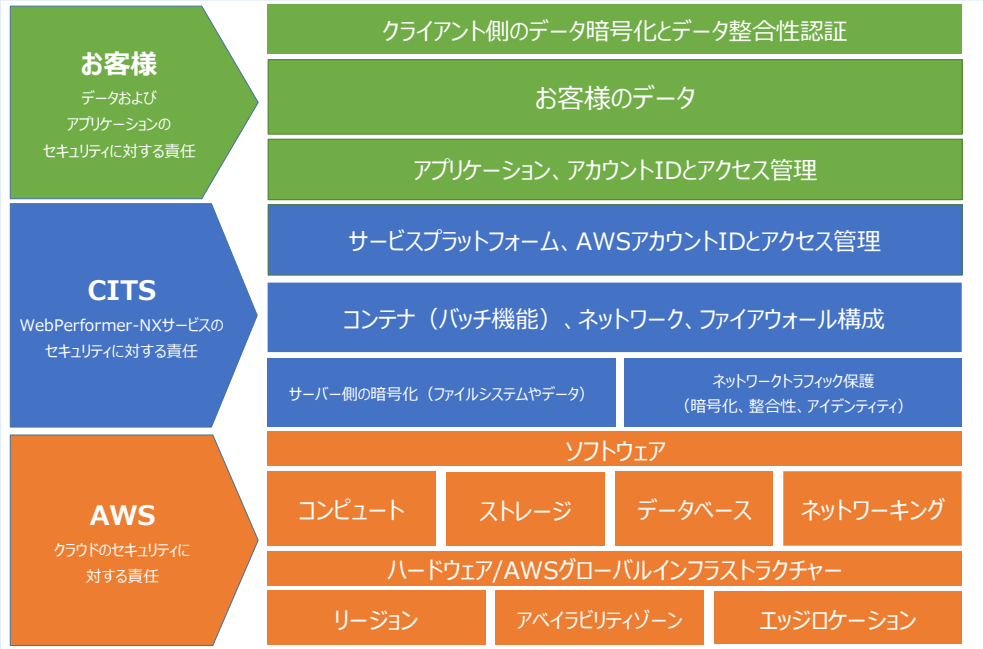
リリース前に、本番用実行環境の設定を確認してください。

- **ユーザ/グループやデータベース、ファイルなど、必要なデータやリソースは登録されているか？**
- **カスタムドメインを利用する場合、設定は行われているか？指定したドメインでアクセスできるか？**
- **IPアドレスでアクセス制限を行う場合、[許可するIPリスト]は正しく設定されているか？**
- **パスワードポリシーをカスタマイズする場合、正しく設定されているか？**
- **送信元メールアドレス（From）をデフォルトのアドレスから変更したい場合、送信元メールアドレス登録と送信元ドメイン登録（※）が完了しているか？**

※送信元ドメイン登録

メールの認証を成功させるため、送信元ドメイン登録が必要です。
登録時にはWP-NXで送信元ドメイン登録後に発行されるCNAMEレコードの情報を、登録した送信元ドメインのDNSサーバーに追加する必要があります。

※WebPerformer-NXにおける責任共有モデルの考え方



リリースと運用

ユーザに提供する最終成果物を実行環境にリリースしましょう。

[実行環境の管理]機能で、各リソースの使用状況や各種ログを確認することができます。

◆ 実行環境の管理

- モニタリング

  モニタリングビューから、下記の情報を確認することができます。
  - ユーザ（登録数/契約数：人、%）
  - リクエスト（使用数/契約数：件数、%）
  - Eメール（送信数/契約数：件数、%）
  - ファイルストレージ（使用量/契約量：MB、%）
  - データベースストレージ（使用量/契約量：MB、%）
  - AIクレジット（利用数/契約数：件数、%）
  - バッチ実行時間（実行時間/契約時間：秒、%）

- ログ

  ログビューから、下記のログを確認することができます。
  - アクセスログ（アクション実行やサインイン成功など）
  - イベントログ（アプリのデプロイ、DB操作など）
  - エラーログ（サインインの失敗、アプリケーションのエラーなど）
  - コンソールログ（アプリケーションのconsoleログ）

以上となりますが、開発時の困りごとなどございましたら、下記までお気軽にお問い合わせください。

E-Mail：WebPerformer-NX担当（wp-nx_info@canon-its.co.jp）

APPENDIX

- フリープランとプレミアムプランとの主な違い
- デバッグデプロイとリリースデプロイの違い
- テストツールの導入
-  WebPerformer-NX 実行環境のシステム構成
- 開発ガイドライン

フリープランとプレミアムプランの主な違い

	無償（Free）プラン	有償（Premium）プラン
実行環境の構成	マルチテナント型	シングルテナント型
最大ユーザ数	5ユーザ	20ユーザ～
最大リクエスト数	8,000リクエスト/月	120,000リクエスト/月～
ユーザグループ管理機能	×	○
IPアドレスによるアクセス制限機能	×	○
カスタムドメイン設定機能	×	○
チーム開発機能	×	○
DB自動バックアップ機能	×	○
バッチ利用オプション機能	×	○
デプロイ可能なアプリケーション数	1	制限なし
DB数	1	複数作成可能
技術情報の提供	・サポートサイトによる技術情報提供	・サポートサイトによる技術情報提供 ・サポート窓口による有人問合せ対応 9:00～17:00（土日、祝日、弊社休業日を除く）
その他	一定期間（90日間）アクセスが無い場合、アプリケーション/データは自動削除される	-

デバッグデプロイとリリースデプロイの違い

console利用例

```
try {
  console.debug(“処理開始");
  const files = await $fn.getUploadFiles();
  await Promise.all(files.map(async (file) => {
    const path = `/uploads/${file.filename}`;
    console.info(“putFile:“+path);
    // 指定パスへファイルを保存
    await $fn.putFile(path, {
      data: file.data,
    });
  }));
} catch (err) {
  console.error("Error ", err);
}
console.debug(“処理終了");
```

デバッグデプロイは、リリースデプロイと以下のような違いがあります。
アプリケーションの動作にも差異があるため、
検証したい内容によりどちらでリリースするかをご判断ください。

デバッグデプロイ利用時の特徴や留意点

① UIエディタで開いている画面が初期表示されます。
② ユーザ操作により、アプリ上でデバッグコンソールを開くことができます。
　console.log()関数等で、デバッグコンソールにメッセージを出力できます
　（実行環境のログファイルにも出力されます）。
③ 認証機能ONに設定していても、無効化されます。
④ $sessionオブジェクトを利用できません。
⑤ ログが即時に反映されます（リリースの場合は1分かかります）。

※バッチアプリケーション、REST APIアプリケーションについては違いは⑤のみとなります。

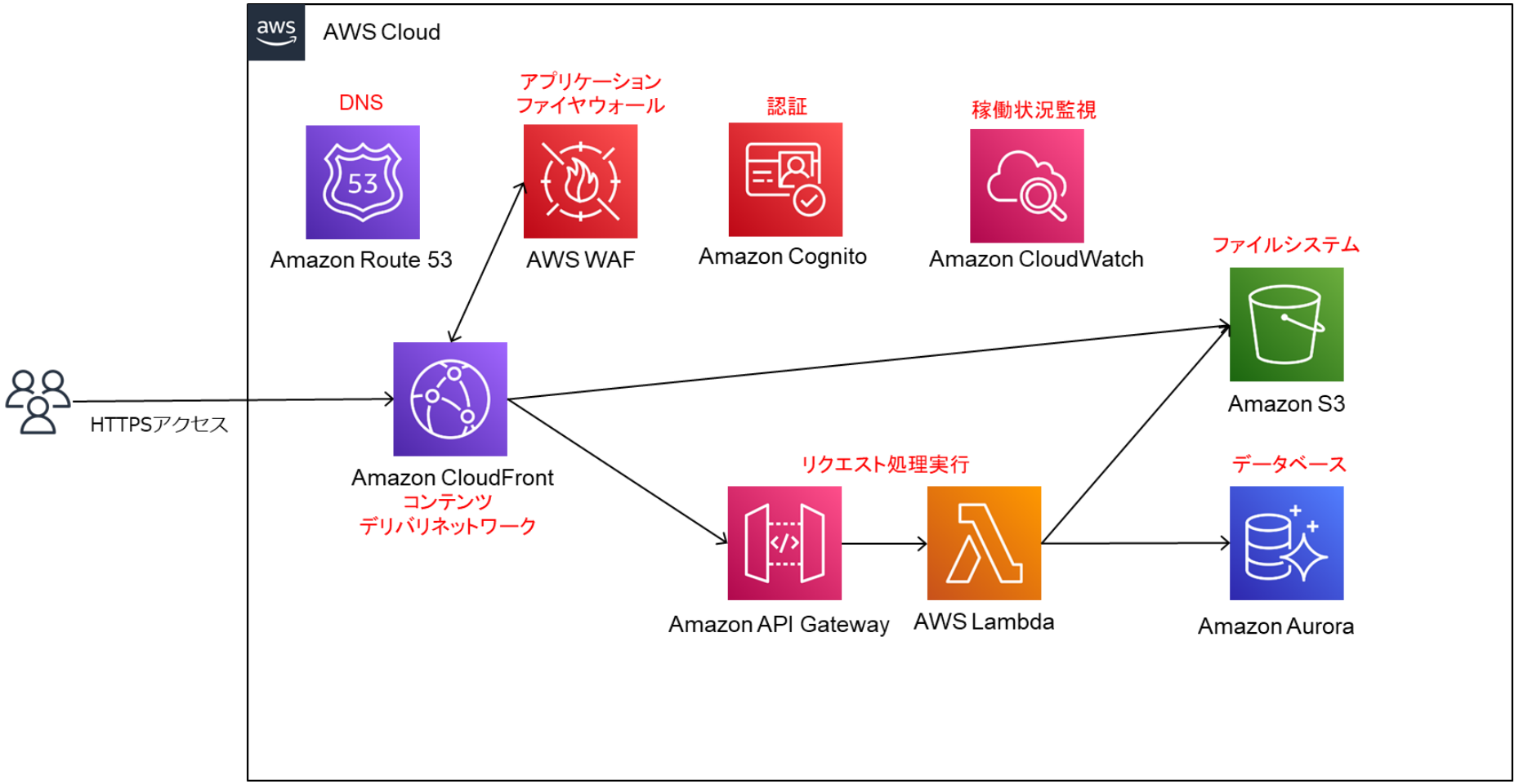
テストツールの導入

WebPerformer-NXで構築したアプリケーションは、Webブラウザ上で動作可能な一般的なWebアプリケーションテストツール/フレームワークであればご利用いただけます。
弊社で稼働実績のあるツールをご紹介します。
（**※動作を保証するものではございません**）

■ Selenide（https://selenide.org/）
開発元：Selenide project
ライセンス：Apache License 2.0

■ Cypress（https://www.cypress.io/）
開発元： Cypress社
ライセンス：MIT（オープンソース版）

WebPerformer-NX 実行環境のシステム構成



※プレミアムプランは、お客様毎に専用環境をご提供します。

フリープランは、マルチテナント環境ですが、ほぼ同一の構成です。

開発ガイドライン

複数の開発者でチーム開発を行う場合、開発ガイドラインを定めて、開発を進めることが望ましいです。
以下に  WebPerformer-NX開発でのガイドライン項目(参考例)を記載します。

1. 用語集
2. 開発体制
3. 開発の流れ
4. ドキュメント
    i. 要件定義書
    ii. 外部設計書、画面遷移図
    iii. 内部設計書
    iv. データベース定義書
5. アプリケーション設定ルール
    i. デフォルトDBの指定
    ii. ファイルパスの取り決め
6. 開発時の命名ルール
    i. アプリID/アプリ名
    ii. UI画面ID/UI画面名
    iii. UIコンポーネントID/UIコンポーネント名
    iv. ユーザ関数名
    v. DB名/テーブル名/列名
    vi. バッチジョブID/バッチジョブ名
    vii. REST API ID/REST API名

    （参考：https://support.canon-its.co.jp/product/web_performer_nx/faq/pk1/?サンプル/命名ルール）
7. JavaScript コーディング規約
8. 変更管理ルール
    i. 成果物の管理方法
    ii. 変更管理内容の記録方法
    iii. リポジトリ機能の運用ルール
9. テストルール
    i. テストケースの作成
    ii. テストデータの管理
    iii. エビデンスの保管
10. 利用ソフトウェア
    i. ブラウザ、開発支援ツール、テストツールなど

Canon

キヤノン ITソリューションズ株式会社