職務経歴書 2023/12/14

概要

バックエンドの開発を中心に、サーバーレスでのアプリケーション構築や、CI/CD の構築、iOS/Android 向けのネイティブアプリ開発、Kintone や intra-mart などの業務プラットフォーム上での UI カスタマイズなど、フルスタックにシステム開発に携わってきました。
また、顧客との折衝やユーザーのヒアリングなども担当し、プロジェクトの立ち上げから、技術選定、要件定義、コーディング、運用までの全ての工程を経験してきました。
数人規模のプロジェクトではありますが、チームリーダーとしてアジャイルな開発、心理的安全性を重視したチームビルディングにも取り組んできました。

以下、これまで携わったサービスです。

- EC サイト
- ネイティブアプリ
- 食品工業メーカーの基幹システム
- リース会社の基幹システム

スキル

- Java
    - 実務経験 5 年
    - フレームワーク Spring, Seasar2
    - 使用バージョンは Java SE 8 まで
    - Spring を使用したアプリケーションの作成を一から経験
    - JSP を使用した動的ページの生成
- PHP
    - 実務経験 3 年
    - フレームワーク Zend Framework, FuelPHP
    - 使用バージョンは PHP 5.6 まで
- JavaScript
    - 実務経験 3 年
- TypeScript
    - 実務経験 2 年
    - アプリケーションの作成を一から経験
- Node.js
    - 実務経験 2 年
    - AWS Lambda で使用
- Dart(Flutter)
    - 実務経験 1 年
    - アプリケーションの作成を一から経験
    - iOS, Android へのアプリリリースも対応
- AWS
    - API Gateway, Lambda, DynamoDB, Cloud Formation, パラメータストア, SAM を使用
    - 実務経験 2 年

    - サーバーレスアプリケーションを構築
- Firebase
    - Cloud Functions, Cloud Firestore, Authentication, Cloud Messaging を使用
    - 実務経験 1 年
- RDB
    - Oracle, MySQL, PostgreSQL を使用
    - 実務経験 10 年
- NoSQL
    - DynamoDB, Cloud Firestore を使用
    - 実務経験 1 年
- CI/CD
    - GitHub Actions を使用して自動テスト、自動デプロイを構築
    - Azure Pipelines の運用、保守
- React
    - 実務経験 2 年
    - Kintone カスタマイズで使用
    - 小さなアプリケーションのため Redux などの状態管理用ライブラリは使用していません
    - 公式のチュートリアルを一通り対応
- Vue.js
    - 実務経験 半年
    - 既存のコードに対して機能追加を行う
- その他
    - Docker、Vagrant、Ansible、Jenkins などを業務で使用
    - チームビルディング
        - 心理的安全性、HRT、コードレビューの導入などを行う
    - スクラムマスター

アウトプット

- Qiita
- GitHub

資格

- 基本情報技術者

職務経歴 詳細

N 社

- 勤務期間: 2021 年 12 月〜
- 雇用形態: 業務委託
- 社員数: 約 250 名

新商品追加プロジェクト

プロジェクト概要

短期で低価格の新車リースを開始。 残存価額を高く設定することで、総リース料を低く抑え、短期利用の新たな客層を取り込む施策です。

既存の商品設計と異なるため、新商品用のマスター作成や見積もり作成機能や新リース料計算、新商品用の契約書印刷機能の作成、車両発注の連携の変更などを行いました。

期間、規模感、役割

- 2023 年/3 ヶ月以内
- 要員 3 名
- 業務アプリに関する部分のチームリーダー
- 他部署へのヒアリング・調整・デモ
- スクラムマスター、開発者

技術

- TypeScript
- Jest
- React
- AWS
    - CloudFormation
    - API Gateway
    - Lambda
    - Systems Manager Parameter Store

使用した外部サービス

- Kintone（業務アプリ）
- AWS（API）
- Slack（処理結果の通知）
- Zapier（メール送信）
- Monday.com（タスク管理）
- Swagger（API 仕様書）

アーキテクチャー

- ドメイン駆動設計
- オニオンアーキテクチャー
- REST API
- アトミックデザイン

取り組んだ課題や成果

企画書を元に業務アプリの要件の策定から携わりました。まずは新商品がどういうものかを理解し、ユースケース図・シーケンス図などの UML を作成。次に業務アプリでの表現方法を考え、セールスや CS など実際に業務アプリを使うユーザーにどうすれば使いやすいか、直感的かをヒアリングし、仕様を確定していきました。 実際の開発タスクは既存のコードへの機能追加でしたが、保守性・可読性の向上を意識しながら都度リファクタリングを行いながら進めました。
実装後はユーザーに動作に関する説明を実施しレクチャーすることで、リリース後も操作に関する問い合わせはな

く、スムーズに業務してもらうことができました。セールスからは違和感なく見積もりを作ることができると評価いただきました。

仕入れ先追加プロジェクト

プロジェクト概要

これまで 1 社から仕入れていたところを新しい仕入れ先に対しても発注できるようにしました。

期間、規模感、役割

- 2023 年/半年以内
- 要員 3 名
- 業務システムに関する部分のリーダー
- 他部署へのヒアリング・調整・デモ
- スクラムマスター、開発者

技術

- TypeScript
- Jest
- React
- AWS
    - CloudFormation
    - API Gateway
    - Lambda
    - Systems Manager Parameter Store

使用した外部サービス

- Kintone（業務アプリ）
- AWS（API）
- Slack（処理結果の通知）
- Zapier（メール送信）
- Monday.com（タスク管理）
- Swagger（API 仕様書）

アーキテクチャー

- ドメイン駆動設計
- オニオンアーキテクチャー
- REST API
- アトミックデザイン
- テスト駆動開発

取り組んだ課題や成果

スクラムを取り入れた開発を行いました。要件の確認、業務システムを使用するユーザーへのヒアリングを行った後、開発チームメンバーで実際に集まってインセプションデッキとプロダクトバックログの作成を行いました。付箋を使ってユーザーストーリーを洗い出したり、ポーカーでの見積もりを行ったりしました。違う意見が出た際はそれ

ぞれの話を聞いてみんなで会話することで、プロジェクト開始時点で全員の認識を揃えていきました。
スプリントの中でスプリントプランニング、デイリースクラム、スプリントレビュー、レトロスペクティブのイベントを実施し、妨げがある際はスクラムマスターとして解消していきました。特にレトロスペクティブに力を入れて、チームの幸福度を上げるためにどんなことに興味があるのかや、チームやタスクに対する意見等をヒアリングし調整していくことで、プロジェクトが終わるまで常に満足度の高いチームでいることができました。ベロシティも安定し、見積もりの精度もかなり高いものになりました。レトロスペクティブに関して記事を書いているので、詳しくはこちらを見ていただければと思います。

バックエンドは、ドメイン駆動設計とオニオンアーキテクチャーを使用して設計し REST API を作成しました。アーキテクチャーの選定理由としては、Kintone をやめるという話があったためプレゼンテーション層やインフラ層に影響されない作りにしたい、正しい要件が誰もわからない状態の機能があったことからコードから業務知識を得ることができるようにしたい、可読性・保守性を上げたい、というものがあります。まずはシステム関連図・ユースケース図・ドメインモデル図・オブジェクト図を作成し、開発しながらフィードバックサイクルを回し、より良いモデリングを行なっていきました。

フロントエンドは Kintone をベースに、React と TypeScript を使って UI をカスタマイズしていきました。その際はアトミックデザインを参考にし、原子・分子・生体の要素に分け、パーツを再利用可能な形で開発しました。

また、レトロスペクティブの中でメンバーからテスト駆動開発の提案があり、開発に取り入れました。スケジュールもある中で全てで実践することは難しかったですが、可能な限り行うようにし、カバレッジを計測しながら最低でも 70% を超えるようにしました。そのおかげでバグの早期発見に繋がったとともに、チーム内での学習意欲が向上し、満足度の向上に寄与しました。

以上の取り組みにより、バグが少なく、保守・拡張しやすいシステムを作ることができました。また、信頼度・満足度の高いチームにすることができ、高いパフォーマンスで開発することができました。

[スクラムの補足]

- 選定理由
  これまで自学しながら自己流でアジャイル開発をやってきましたが、一度フレームワークを使ってやってみようと思い、スクラムを選択しました。
  経験者がいないこともあり、あまり厳密ではなく自由度の高いフレームワークの方が推進しやすいと考え、他のフレームワーク（例えばエクストリームプログラミング）ではなくスクラムにしました。
- 所感
  - 良かったこと
    役割・イベント・成果物が明確に定義されているので始めやく、記事や書籍が多数出ているので学習がしやすかったです。書籍では『SCRAM BOOT CAMP』と『スクラム』から基礎的な知識を得ました。
    また、チームとしての一体感、見積もりの精度がかなり向上したこともスクラムを取り組んだ成果です。
  - 悪かったこと
    初めて組むメンバーだったので、プロジェクトの初めの段階でいつ頃終わるのか不明確だったことが挙げられます。
    また、学習意欲の高い人・スクラムをやりたい人でチームが構成されていたので良かったですが、もし学習意欲のない人や自立できていない人がメンバーにいる場合はうまく回らないように感じました。

[スクラムマスターの補足]

- 所感

    - 良かったこと
      デイリースクラムやレトロスペクティブなどのイベントを通して、チームとしてまとまっていく過程にとても達成感がありました。
    - 悪かったこと
      開発者としての役割も担っていたので、タスクの切り替えに苦労しました。スクラムマスターとしての時間、開発者としての時間を区切ってタスクの管理を行ないました。

[ドメイン駆動設計とオニオンアーキテクチャーの補足]

- 選定理由
  取り組んだ課題や成果に記載している通りですが、プレゼンテーション層、インフラ層に該当する Kintone を辞めるとの話があったので、それらに依存しない息の長いシステムにしたったことが一番の理由です。
  クリーンアーキテクチャーという選択もありましたが、言葉の定義が直感的で分かりやすいオニオンアーキテクチャーを選択しました。
- 所感
    - 良かったこと
      これまで以上にユーザーへのヒアリングを行なったことで、詳しい業務知識を得ることができました。
    - 悪かったこと
      メンバー全員で学習する時間を作りました。また、コードに落とし込む際に意見しあって、しっくりくるものができるまで何回もフィードバックサイクルを回したので学習コスト・初期開発コストは高いと感じました。

パッケージ管理導入プロジェクト

プロジェクト概要

Kintone をベースに業務システムを作っており、アプリごとにプロジェクトを分けている環境で、 複数のアプリで必要な共通のビジネスロジックをアプリ毎に書くという DRY 原則に反した作りになっていました。そのため、ビジネスロジックの変更による影響範囲が大きくなってしまっていました。そこで、共通化できる処理をライブラリとして提供できるよう GitHub Packages を導入しました。

GitHub Packages を選んだ理由は以下です。

- プライベート NPM として公開したい
- 導入コストが低い

小さく始めて、軌道に載ったらスケールアップさせる方針で進めました。
初回の機能は以下です。

- アプリの型定義
- データの CRUD
- 金額計算処理

期間、規模感、役割

- 2022 年/2 週間以内
- 要員 1 名
- 問題提起から技術選定、コーディング、テスト、その後の保守運用の全てを担当

技術

- GitHub Packages
- GitHub Actions
- TypeScript
- Jest

取り組んだ課題や成果

リリース作成時に GitHub Actions で自動デプロイするように対応しました。
また、広く使用してもらう関数のため、必ずインターフェース仕様書とテストコードの作成を行なっています。
ライブラリから外部の Web API に通信している機能については Mock Service Worker でネットワークレベルでのモックを作成しテストを行っています。

業務の中で抱えている問題を自ら提起し、解決に至りました。
リポジトリの運用ルールも定め、私が責任を持って管理しています。

GitHub Packages を使用した社内向けライブラリの作り方について、詳しくはこちらの記事に記載しています。

会計システム改修プロジェクト

プロジェクト概要

会計システムに問題があり、決算の数字が正しく出力できていませんでした。そこで経理部にヒアリングしながら正しく数字を出せるようにシステムの改修を行いました。

期間、規模感、役割

- 2022 年/3 ヶ月以内
- 要員 2 名
- ユーザーへのヒアリングから設計、コーディング、テスト、その後の保守運用の全てを担当

技術

- TypeScript
- JavaScript
- Jest
- React
- Vue.js

使用した外部サービス

- Kintone（業務アプリ）
- Slack（処理結果の通知）
- Monday.com（タスク管理）

取り組んだ課題や成果

まずは原因となるお金のズレを調査するところから始めました。経理が把握しているおかしなデータをヒアリングし、料金計算のロジックを見直すことでバグを見つけ出すという地道な作業を行いました。例えば、特定の条件下における消費税計算時の端数処理がおかしいなどの問題がありました。

次に会計業務に必要な機能の追加を行いました。過剰入金・部分入金の消込、消込の取り消し、入金データ登録時の自動消込、口座振替結果の自動登録、利用者への口座割り当て機能を対応しました。

アジャイルで開発を進めました。1日1回プルリクエストをだせる粒度でタスクを細かく分割し、毎週リリースを行いました。それにより進捗が明確化し経理部担当者から進捗確認の連絡がくることはなく、コミュニケーションをスムーズに行うことができました。また、不具合を出すことなく進めることができました。

以上の取り組みにより、決算の数字を正しく出力できるようになりました。
長年の問題だったこと、経理部の業務効率が上がったことから、その月の MVP 、年度末に行われる総会での MVP 候補に選ばれました。

業務システムのデータフロー見直しプロジェクト

プロジェクト概要

サービスに申し込まれてから契約を管理するまでの業務システムに問題があり、手動でリカバリしながら運用を行なっていました。そこで申し込みから契約情報の管理までを一貫して業務システムで行えるようにフローの見直しとシステムの改修を行いました。

期間、規模感、役割

- 2022 年/半年以内
- 要員 4 名
- 要件定義、技術選定、設計、コーディング、レビュー、テスト、その後の保守運用を担当

技術

- TypeScript
- JavaScript
- Jest
- React
- AWS
    - CloudFormation
    - API Gateway
    - Lambda
    - Systems Manager Parameter Store

使用した外部サービス

- Kintone（業務アプリ）
- AWS（API）
- Slack（処理結果の通知）
- Zapier（メール送信）
- Monday.com（タスク管理）

取り組んだ課題や成果

Kintone をベースに業務システムが構築されていましたが、各アプリ間の連携や処理のタイミングに問題があり契約情報を管理するまでの処理で人が目で見てデータを更新するなど手動での運用が必要なフローが多々ありました。
そこでアプリ一つに対して一つの責務になるようにアプリを分割したり、Webhook 機能を使い Lambda で他アプリ

に自動でレコードを作成したりするなどの対応を行いました。そうすることで人の手を介さずに正確に契約情報の作成までをシステム化することができるようになりました。
また、システムで問題が起きたときは Slack に通知し、すぐにエラーを検知できるようにしています。

プロジェクトの進め方としては、私の提案でアジャイルを採用しました。
アジャイル開発未経験者ばかりだったので、まずはチームメンバーでアジャイルの学習を行いました。まずは『アジャイルサムライ』を全員で読み、全員の意識を合わせるところから始めました。
インセプションデッキ、エレベーターピッチ、やらないことリスト、トレードオフ・スライダーなどの資料の作成を行い、プロジェクトの全体像を描きメンバー全員が同じバスに乗れるようにしました。
また、タスクを細かく分割し、隔週でリリースを行うことで、一度のリリースによる影響範囲が少なく安心して行うことができました。

申し込みから契約情報の管理まで一貫してシステムで行えるようになり、データの整合性も取れるようになったことで生産性に大きく貢献することができました。

[アプリの責務に関する補足]
一部例を記載します。審査用のアプリで発注情報も含めたリース会社とのやりとり全てを行なっていました。なので審査用のアプリから発注に関する役割を剥がして、新規で発注アプリを作成し、そちらに機能を移しました。

[アジャイルに関する補足]
問題が大きすぎてゴールが明確化していない状況でプロジェクトが進み出しました。参画して日が経っていないメンバーばかりだったこと、業務アプリのオペレーションが複雑で奇想天外だったこともあり、具体的に何が問題でどう解決して、それはどれくらいで終わるのか全く把握できない状況です。そこで「不確実で混乱した環境に対処し、最終的に成功すること」を目的とするアジャイル開発を提案しました。チーム内からはやってみよう！とポジティブな意見をいただき、本を参考に自分たちで学びながら、やりやすい方法で進めました。細かな手法はあるかとは思いますが、私たちが重視した点は以下の 2 点です。

- タスクを細かく分解すること（1 日 1 プルリク）
- 早い段階でフィードバックを得ること

2 週間単位でイテレーションを回し、月曜日リリース（デプロイ）、その前の週後半でショーケースというスケジュールで進めました。

DynamoDB データ洗い替え

プロジェクト概要

メンテナンスプランの利用可能料金の見直しのため、DynamoDB のデータの更新を行いました。

期間、規模感、役割

- 2022 年/1 ヶ月以内
- 要員 1 名
- 要件定義、設計、コーディング、レビュー、テスト、リリースを担当

技術

- Amazon DynamoDB
- TypeScript
- Jest

取り組んだ課題や成果

DynamoDB の特性を理解しならが、該当データを取得、更新するバッチを作成しました。

C 社

- 勤務期間: 2016 年 7 月〜 2021 年 8 月
- 雇用形態: 正社員
- 社員数: 約 10 名

健康管理アプリ開発プロジェクト

プロジェクト概要

医療や健康に関するデータを記録し、Android、iOS 向けの医師やトレーナーと共有できる PHR（パーソナルヘルスレコード）アプリを開発しました。

以下のような機能がある、10 画面程度のアプリです。

- ログイン
- データの CRUD
- データを医師に共有する
- データのグラフ表示
- 運営からの通知
- サブスクリプションの契約

期間、規模感、役割

- 2021 年/1 年以内
- 要員 2 名（フロントエンド 1 人、バックエンド 1 人）
- フロントエンドの技術選定、設計、コーディング、レビュー、テスト、その後の保守運用を 1 人で担当。
- Android、iOS アプリの申請・リリースも担当。

技術

- Dart
- Flutter
- Postgresql
- Cloud Firestore
- Firebase
    - Authentication
    - Crashlystics
    - Cloud Messaging（FMC）

取り組んだ課題や成果

社内で初めてのネイティブアプリ開発ということもあり、技術選定から担当しました。
1 人で開発するとあって工数をかけることができないため、クロスプラットフォームである Flutter を選択しました。
React Native も検討しましたが、以下を理由に Flutter にしました。

- 開発工数を抑えられるクロスプラットフォームの言語
- 言語に関する記事が多く、今後伸びていきそう

状態管理は、Stateful Widget Pattern で対応しています。複雑な状態管理が不要なシンプルなアプリであること、学習コストの少ないことが理由です。
ログイン機能等は Firebase を使用しています。
アプリ内課金や Android、iOS アプリの申請・リリースも対応しました。
初めて、かつ周囲に知見者がいなかったので調べながら、iOS アプリの申請では担当者とやりとりしながらリリースしました。
特に課金に関しては、当初 Stripe による決済システムを実装していましたが、In-App-Purchase (アプリ内課金) でないとリジェクトされてしまい苦労しました。
紆余曲折はありましたが、一から新しいサービスを作ることができました。

食品メーカー基幹システムのリプレイス

プロジェクト概要

食品メーカーの基幹（受注・製造・出荷・経理）システムをリプレイスしました。
また、流通 BMS に対応するなど、ユーザーからのヒアリングを元に必要となる仕組みを導入しました。

期間、規模感、役割

- 2016 年/2 年以上
- 要員 4 名
- 主に受注部分の設計、コーディング、レビュー、テスト、その後の保守運用を担当
- ユーザーマニュアルなどのドキュメントの作成、リリース後のユーザーサポートも行う

技術

- Java（Spring）
- JavaScript
- PostgreSQL
- Oracle
- Apache
- intra-mart

取り組んだ課題や成果

20 年ほど前に Delphi で作られた食品メーカーの基幹システムを Web システムとしてリプレイスしました。

D 社

- 勤務期間: 2013 年 10 月〜 2016 年 6 月
- 雇用形態: 正社員
- 社員数: 約 1200 名

サーバー分離対応

プロジェクト概要

WEB サーバーに入っていた決済確定処理を分離しました。
12 種類ある代行会社を経由した決済手段のうち、10 種類は対応完了。

期間、規模感、役割

- 2016 年/半年以内
- 要員 1 名
- 開発、単体テスト、結合テスト、運用

技術

- PHP
- HTML
- Javascript
- MySQL
- Linux
- Apache

業務内容

- インフラ部への作業指示依頼
- 代行会社へのドメイン変更依頼
- 検証環境の作成（バーチャルホストを設定し同サーバーにて検証を実施）
- STG 環境にて代行会社検証環境と連携し検証を実施

決済手段ビットコイン追加対応

プロジェクト概要

決済手段にビットコインを追加しました。

期間、規模感、役割

- 2015 年/半年以内
- 要員 10 名
- 開発、単体テスト、結合テスト、総合テスト、運用

技術

- PHP
- HTML
- Javascript
- MySQL
- Linux
- Apache

業務内容

- 申込み、決済確定処理の開発、テストを一人で担当
- 決済代行会社との調整

- 障害発生時の運用フローを作成

経理システム　予約販売対応

プロジェクト概要

予約販売開始により売上データの持ち方、経理システムへの計上タイミングを変更しました。

期間、規模感、役割

- 2015 年/1 年以内
- 要員 2 名
- 設計、開発、単体テスト、結合テスト、運用

技術

- PHP
- HTML
- Javascript
- MySQL
- Linux
- Apache

業務内容

- バックエンドの設計、開発、テスト
- 経理部との調整
- 他事業部エンジニアとの調整

ポイント統合対応

プロジェクト概要

複数存在する金種を 1 つに統合。
プラットフォーム事業部の機能を API 化。
FuelPHP を導入し、リファクタリングを実施。

期間、規模感、役割

- 2014 年/1 年以内
- 要員 50 名
- 開発、単体テスト、結合テスト、運用

技術

- PHP
- HTML
- Javascript
- MySQL
- Linux

- Apache

業務内容

- 経理システム、売上日報の設計、開発、テストを一人で担当
- 売上日報をリファクタリング
- カスタマーサポートシステムをリファクタリング

EC サイト、基幹システム 短期改修案件

プロジェクト概要

プラットフォーム事業部にて改修対応を実施しました。 対象は自社 WEB サイト、経理システム、カスタマーサポートシステム、売上報告システムなど。

期間、規模感、役割

- 2013 年/断続的に
- 要員 50 名
- 開発、単体テスト、結合テスト、運用

技術

- PHP
- HTML
- Javascript
- MySQL
- Linux
- Apache

業務内容

- 不具合改修
- 機能追加
- データ抽出
- 経理システムの開発環境作成

S 社

- 勤務期間: 2011 年 4 月〜 2013 年 10 月
- 雇用形態: 正社員
- 社員数: 約 160 名

EC サイト　 Google Search Appliance 導入

プロジェクト概要

Google 社が提供する検索システム Google Search Appliance の導入検討にあたり、フロントエンドシステムの開発を行いました。

Solr との AB テストを実施。

期間、規模感、役割

- 2013 年/3 ヶ月以内
- 要員 6 名
- 基本設計、詳細設計、開発、単体テスト、結合テスト、総合テスト、リリース、運用、検証

技術

- Google Search Appliance
- Oracle
- MySQL
- Linux
- Apache
- Jboss

業務内容

- 開発環境の整備
- フロントエンドシステムの設計、開発、テスト
- 協力会社 3 名のマネジメント
- Google Search Appliance の運用業務
- AB テストの結果検証、報告

EC サイト　Solr 検索改善

プロジェクト概要

検索経由売上向上のため、システム施策の提案。人気ワードの設置や Solr の重みづけ、同義語辞書、ユーザー辞書の改善を実施しました。

期間、規模感、役割

- 2013 年/3 ヶ月以内
- 要員 4 名
- 企画、基本設計、詳細設計、開発、単体テスト、結合テスト、総合テスト、リリース、運用、検証

技術

- Solr
- Java
- JSP
- HTML
- CSS
- MySQL
- Linux
- Apache
- Jboss

業務内容

- 施策の企画、提案
- フロントエンドシステムの設計、開発、テスト
- Solr の運用業務

カスタマーセンターでのお問い合わせ業務

プロジェクト概要

クリスマスの繁忙期で人手不足であったこと、またカスタマー業務の理解も兼ねて、開発部全体で開発業務の合間にカスタマーセンターでのお問い合わせ業務を行いました。

期間、規模感、役割

- 2012 年/1 ヶ月以内

業務内容

- 問い合わせメールの対応
- 返品処理、キャンセル処理など

EC サイト　ファンクラブ会員限定商品販売

プロジェクト概要

大手芸能事務所のオフィシャルサイトの顧客を取り込むため、会員限定で利用できるサービスの立ち上げを行いました。

期間、規模感、役割

- 2012 年/1 ヶ月以内
- 要員 1 名
- 要件定義、基本設計、詳細設計、開発、単体テスト、結合テスト、総合テスト、リリース

技術

- Java
- JSP
- HTML
- CSS
- MySQL
- Linux
- Apache
- Jboss

業務内容

- 要件定義書作
- バックエンドシステムの設計、開発、テスト

- フロントエンドシステムの設計、開発、テスト
- オーダーシステムの設計、開発、テスト
- 導入手順書作成

EC サイト　アフィリエイト

プロジェクト概要

アフィリエイト事業者との新規契約に伴い、商品データ送信システム、フロントシステム、成果報告システムの開発を行いました。

期間、規模感、役割

- 2012 年/1 ヶ月以内
- 要員 2 名
- 要件定義、基本設計、詳細設計、開発、単体テスト、結合テスト、総合テスト、リリース

技術

- Java
- JSP
- HTML
- CSS
- MySQL
- Oracle
- PL/SQL
- Linux
- Apache

業務内容

- アフィリエイト事業者との打合せ
- 要件定義書作成
- フロントシステムの設計、開発、テスト

EC サイト　ダウンロードゲーム販売

プロジェクト概要

ダウンロードゲームの販売に伴い、在庫投入システムからオーダーシステムの開発を行いました。

期間、規模感、役割

- 2012 年/3 ヶ月以内
- 要員 4 名
- 基本設計、詳細設計、開発、単体テスト、結合テスト、総合テスト、リリース

技術

- Java

- JSP
- HTML
- CSS
- MySQL
- Oracle
- PL/SQL
- Linux
- Apache

業務内容

- オーダーシステムの設計、開発、テスト
- メールシステムの改修

EC サイト　グループネット一本化プロジェクト

プロジェクト概要

グループ会社サイトの統合に伴うシステム開発を行いました。

期間、規模感、役割

- 2011 年/1 年以内
- 要員 35 名
- 基本設計、詳細設計、開発、単体テスト、結合テスト、総合テスト、リリース

技術

- Java
- JSP
- HTML
- CSS
- Oracle
- MySQL
- PL/SQL
- Linux
- Apache
- Solr

業務内容

- フロントエンドの設計、開発、テスト
- 協力会社 3 名のマネジメント
- プロジェクト用開発環境の構築