＜フレームワーク　課題＞
①Spark Framework
概要：マイクロフレームワークの一種。
　※マイクロフレームワーク・・・本質としては、軽量フレームワークの延長で、現時点で
　　得られる最軽量のJava開発手法。
　　集中的で迅速なソリューションを生み出す事で、
　　開発プロセスを必要最低限の内容に絞り込む。
特徴：＜メリット＞
　　シンプルなプログラミングや迅速なシステム開発を行う事ができる。
　　膨大なアノテーション記述や設定ファイルが不要な構成となっており、
　　開発者の負担が大幅に軽減できる。

　　＜デメリット＞
　　必要最低限の機能しか実装されていない為、大規模Webシステムの構築には、
　　不向き。
　　（中～小規模のWebシステムであれば問題なし。）

②JSF
概要：2004年に開発されたJavaベースのWebアプリケーション。
　　JavaEEの仕様にも採用されているJavaの標準フレームワーク。
　　MVCモデル採用のフレームワーク。コンポーネントベースフレームワーク。
　　※コンポーネントベースフレームワーク・・・
　　　表示する画面に対応するサーバーサイドのクラス（バッキングBean）によって
　　　処理が行われる。
特徴：MVCモデルに準じ、Model層（ビジネスロジック）とView層（入出力画面）の開発を
　　明確に区別している。
　　MVCモデルのViewには、JSPで実装されるケースが多いが、
　　JSFは「XHTML」というXML形式のHTMLで実装される。
　　JSPはサーブレットの処理なしでは、正しく表示されないが、XHTMLの場合はブラウザで
　　そのまま表示される為、デザインの確認がしやすい。

③JUnit
概要：Javaで開発されたプログラムにおいてユニットテスト（単体テスト）の自動化を
　　行う為のフレームワーク。
特徴：＜メリット＞
　　一度作成すれば、素早くテスト可能であり、その後はテストコードを標本とする事で、
　　バグ修正が容易となる。
　　テストコードを見れば、仕様が一目瞭然となり、また誰が行っても同じテストを行える。
　　独自のテストコードによるテスト作成の手間が省ける。

　　＜デメリット＞
　　仕様変更ごとにテストコードを作り直さなければならない。
　　テストコード作成に時間がかかる。