入金確認の手順まとめ

入金確認は以下の３つのステップに分けて行います。

ステップ１：入金メール確認

まず、ユーザーの指定されたメールのサーバーに５分１回に、新しいメールが来たかどうかをチェックする。

この時、ユーザーのメールアドレスとパスワードがいる。

ユーザーメールに関する情報まとめ

①メールサーバー：例、mail.mamol.co.jp

②ユーザー名（メールアドレス）：例、[kiku@mamol.co.jp](mailto:kiku@mamol.co.jp)

③ユーザーアカウントのパスワード：例、123456

技術として、pythonのimaplibモジュールを使う。

ただし、メールサーバーがimaplibからの接続が許可されるかどうかチェックする必要がある。

ソースサンプルはcheckmail.pyを参照

5分ごとで、メールサーバーから、一つのメールを取り出して、発信元のアドレスとサブジェクトをチェックする。

特定の銀行からの入金確認のメールがあれば、すぐ、この時点で、ブラウザの銀行入金確認処理を行い、このメールの番号を入金確認データーベースに書いとく、ブラウザの銀行入金確認が終わったら、すぐ、次の受信メールをチェックにいく。

入金確認のメールじゃなければ、次の番号のメールをチェックに行く、メールがなければ、この時点で、スクリピットを5分スリープさせて、5分あとに、また上記のステープを繰り返す。

ステップ２：ブラウザの銀行入金確認

（１）ログイン画面

ユーザーの銀行情報をデータベースに保存されたという前提で、

ブラウザを開いて、銀行のホームページのurl（①）を導入し、ページをブラウザ上で開く。

DBのユーザー名（あるいは、カード番号）とパスワードをページ上の適当のinputフィールドに入力し、ログインボタンを自動的にクリックさせる。

その後、結果を判断する。

ログインが失敗したら、エラーログに書いて、このメッセージを管理員にメールを送る。

銀行のログイン画面に、特別の認証方法があるかどうかを確認する必要がある。

（２）入金確認画面

銀行の入金確認画面を開いて、このページの中に、入金情報を洗い出し、DBに書きこむ。

入金情報がなければ、エラーログに書いて、このメッセージを管理員にメールを送る。

銀行の入金確認画面にどんな情報が掲載されているかを確認する必要がある。

（３）ブラウザを閉じる

技術として、pythonのsplinterモジュールを使う。

注意：各ページを完全にロードさせるために、充足の待ち時間を保証する必要がある。

ソースサンプルはsplinter.pyを参照

ステップ３：csvファイルの生成とアップロード

DBに保存された入金情報に基づいて、csvファイルの生成する。

ｃｓｖファイルの形式を確認する必要がある。

ｃｓｖファイルの生成するために、DBに十分のデータをのせる必要がある。

生成されたcsvファイルを管理画面にアップロードする。

管理システムのURL。