ウェブアプリケーションの作成

名古屋大学情報学部コンピュータ科学科 3 年
堀内颯太 (学籍番号 : 101730331)
horiuchi.sota@e.mbox.nagoya-u.ac.jp

実験日 2019 / 5 / 30, 6 / 6

目次

1　はじめに

1.1　概要

本レポートでは、WWW サーバ上で SQL データベースと協調することで動作する WEB アプリケーションをグループで作成する。WEB アプリケーションは Python の WSGI で稼働させる。本レポートでは、WEB アプリケーションを作成を行い、その総括をする。

1.2　実験機器

本レポートの実験では以下の機器を使用した。

- ルータ
- WWW サーバ
- PC
- PC 切り替え機
- モニタ
- キーボード・マウス
- Ethernet クロスケーブル
- Ethernet ストレートケーブル

ルータ (以後 machine1)、WWW サーバ (以後　machine 2)、PC(以後 machine 3) は PC 切替器を介してキーボード及びマウスと接続した。キーボードは PC 切替器のキーボード用端子に、マウスは PC 切替器のマウス用接続端子に接続した。PC 切替器と machine 1 及び machine 2、machine 3 は RGB ケーブルと USB ケーブルを用いて接続した。PC 切替器とモニタを RGB ケーブルを用いて接続した。Ethernet クロスケーブルを用いて machine 1 と machine 2、machine 1 と machine 3 の LAN インターフェース間を直接接続した。machine 1 と上位ネットワークは HUB を介して接続した。

1.3　データベース

データベースとは多種多様な使用目的を考慮して規則的に保存されたデータの集まりのことを指す。このデータは DBMS(データベース管理システム) によって使用される。データベースシステムには以下の五つの特徴が存在する。

1. プログラムの独立性
   データベースの仕様変更が発生してもプログラムを変更する必要がない場合が多い
2. データの非重複
   データベースとして情報を一元管理するためデータに重複がない
3. 同時処理
   データベースに同時に複数人のユーザがアクセスした場合もデータベース管理システムが制御するため、データの書き込みや削除を同時に行うことができる
4. データの機密性

アクセスをデータベース管理システムで制御できる

5. データの障害回復
   データベースに何らかの障害が発生してもデータベース管理システムがこれを回復する手段を持っている

データベースのデータモデルには、構造型と非構造型が存在する。構造型データベースには階層型データベースとネットワーク型データベースがある、非構造型にはリレーショナル型データベースとオブジェクト型データベースがある。

- 構造型データベース
  データの各属性をひとつのノードとし、それらを各ノードで親子関係で結び付けることにより、データを表現するモデル。
  - 階層型データベース
    全体として木構造をなし、各ノードは親がただひとつしか持たない。このため、1 ノードで親が複数になりうる場合、重複するノードができるという欠点がある。
  - ネットワーク型データベース
    全体として網目構造をなし、各ノードが複数の親レコードを持つことができる。ノードの重複は無くなるが、親子関係が複雑に入り組んだ構造になりやすい。
- リレーショナル型データベース
  各データは 2 次元の表として表現する。複数の表のデータを関連付けることで、すべてのデータをひとつの巨大なデータベースとして表現する。構造型と比較して、容易に巨大なデータベースを構築することができる。現在利用されているデータベースのほとんどのモデル。
- オブジェクト型データベース
  オブジェクト指向設計をデータベース設計に適用したモデル。複雑なデータ構造に適す。パフォーマンスの低さとスケーラビリティ不足のため、ほとんど普及していない。

[1]

1.4 SQL

SQL とは Structured Query Language(構造化問合せ言語) のことで、データベースの定義や操作などを実現するためのデータベース言語の一つ。想定するデータベースの構造はリレーショナル型データベースであり、ISO 及び JIS において規格化されている。現在ではその機能的な使い安さから、リレーショナル型データベースの事実上の標準として位置づけられている。

リレーショナル型データベースを管理するのはリレーショナル型データベース管理システムであり、SQL はその管理システムを制御するデータベース言語である。SQL は主に以下の 3 つの機能を有している。

- データベース定義
  データを格納すべき表の定義や複数の表を関連づけるための規約や制約、データベースに必要な機密保護の宣言がある
- データベース操作
  表に対するデータの登録や修正、削除。複数の表の結合や、ビュー表の作成といった集合操作、表に含

まれたデータの検索がある

- トランザクション管理
  回復や同時実行のための最章単位として保証される一連の処理の操作がある

[1]

1.5 WEB アプリケーション

WEB アプリケーションとは、ブラウザで利用可能なアプリケーション・サービスのことである。クライアント側のブラウザがインタフェースとなり、サーバ側のアプリケーションサーバなどのプログラムと互いに通信をおこなうことでサービスを実現する。クライアントは URL を指定し、サーバへのリクエストを送信する。サーバは、リクエストに応じて事前に設定した処理をおこないそれをレスポンスとしてクライアントに送信する。ブラウザはサーバから受け取ったレスポンスに基づいて画面を構築し、クライアントはまた、そこに入力をし、サーバへのリクエストを発行することで WEB アプリケーションの機能を実現する。

2 実験: SQL データベースと連携した Python WSGI WEB アプリケーションの作成

2.1 概要

2.1.1 実験概要

本実験では、ログイン機能を備えた SQL データベースと連携した Python WSGI WEB アプリケーションの作成を行う。作成したアプリケーションは machine2 の WWW サーバ上で起動させる。

2.1.2 アプリケーション概要

本実験では、Twitter を模した WEB アプリケーションを作成した。本アプリケーションでは、ユーザは自分の ID とパスワードを設定し、ログインをすることで、アプリケーションを利用することが出来る。各ユーザは文章を投稿できる他に、検索ページから他のユーザの投稿も閲覧することができる。また、フォローフォロワー機能も搭載しており、フォローをすることで、そのフォローリストからフォローしている人の ID を見ることが出来そのページからフォローしている人のマイページに遷移することで、その人の投稿の内容を閲覧することができる。

2.1.3 アプリケーションの仕様

本アプリケーションでは、データベースを使用した。また、ユーザは 4 つのページに遷移することができる。大きく分けて login、userpage、userlist、timeline の 4 つページに遷移することができる。

データベース

データベースのテーブルには USER_TABLE、FOLLOWING_TABLE+USERNAME、FOLLOWER_TABLE+USERNAME、TWEET_TABLE、SESSION_TABLE を用意した。FOLLOWING_TABLE+USERNAME、FOLLOWER_TABLE+USERNAME はその USERNAME は動的に作成することが出来、USERNAME に指定したフォローもしくはフォロワーの TABLE になる。USER_TABLE には name,pass のフィールドを作成しユーザのログイン ID とそのパスワードを保存する。このパスワードは user モジュールにて hash 化された状態で保存されている。FOLLOWING_TABLE+USERNAME、FOLLOWER_TABLE+USERNAME にはそれぞれ FOLLOW と FOLLOWER のフィールドを作成し、USERNAME に該当するユーザのフォローフォロワーの関係を保存する。TWEET_TABLE には content、user、data のフィールドを作成し、投稿内容と誰が投稿したのかとその日時を保存する。SESSION_TABLE には token、name のフィールドを作成し cookie にセットする token の値とそのユーザ名を保存する。これらデータベースに関する操作の CGI に対する命令をそれぞれ TABLE ごとに異なったクラスにて実行出きるよう抽象化を行い、CGI に実際に命令する部分もまたクラスによりモジュール化した。

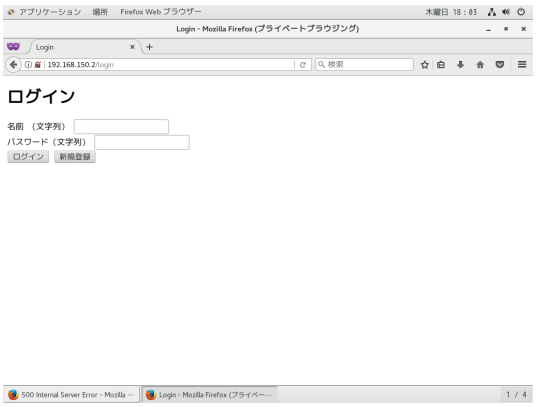

図 1　login 画面

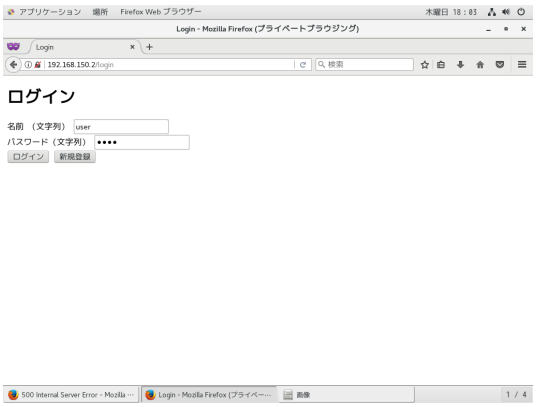

図 2　login フォーラムに文字を入力後

login ページ (担当箇所)

login ページではまず HTTP_COOKIE の確認を行い、クッキーの token を受け取り、session にてデータベースとの照合を行い、name がかえされた場合には userpage にリダイレクトする。エラーが発生した場合は、token の値がないもしくはその値では login できない (つまり過去に session の削除が行われた) ので login ページにそのまま滞在する。

図 1 の login 画面にてユーザは名前とパスワードの入力を行う。図 2 は実際に文字を入力したあとの画面である。login か新規登録どちらかを押すことで login もしくは singin を行う。これらのフォームは post にてリクエストを送信する。アプリケーション側で、リクエストメソッドが POST であった場合、その内容を parse して POST された内容を判定する。どちらか、または両方の text ボックスが空文字列であった場合はどのボックスが空であったかのエラーを出力する。図 3 はパスワードを入力せずにボタンを押した時の画面と図 4 はパスワードと名前両方を入力せずにボタンを押した場合である。

両方の入力があった場合はログインもしくは新規登録どちらのボタンが押されたかを判別して、それぞれの実行をする。ログインの場合では User テーブルの name と pass が異なるか、name が登録されていない場合例外処理をする。その際、セキュリティの観点からどちらのエラーであるかは出力しないことにした。ログイ

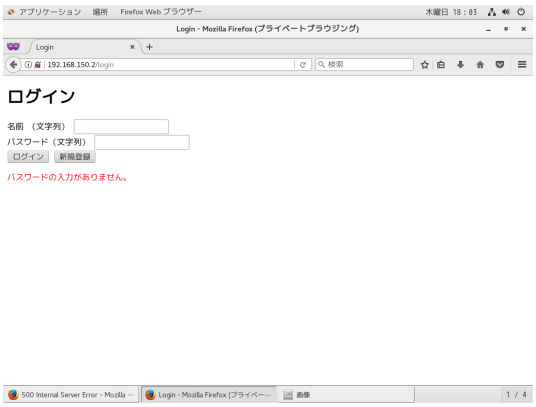

図 3　パスワードを入力しなかった場合

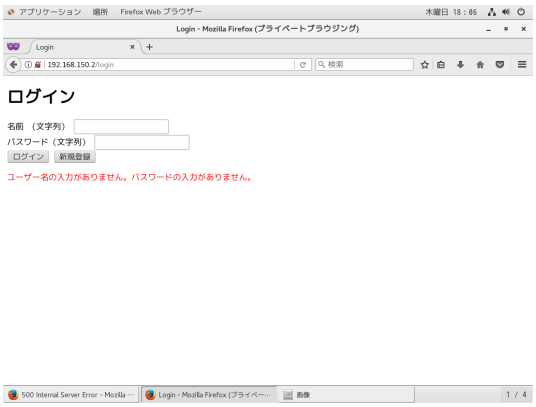

図 4　パスワードと名前を入力しなかった場合

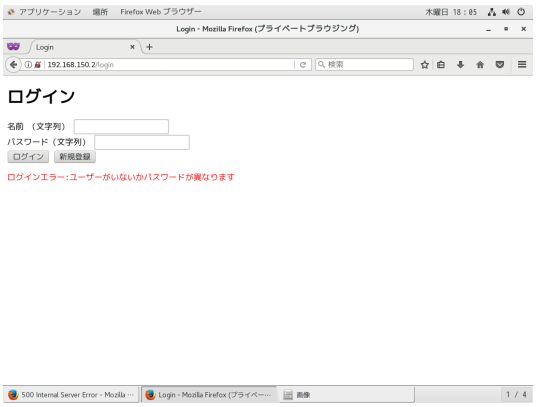

図 5　ログインに失敗した場合

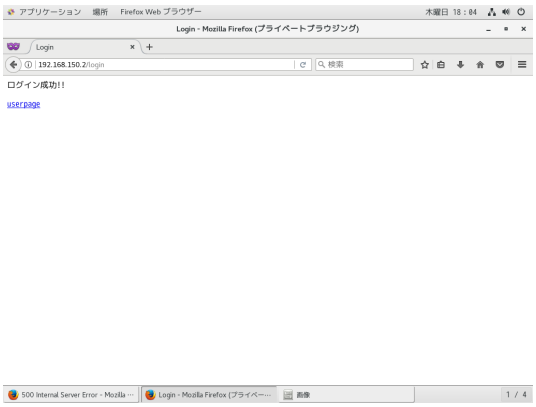

図 6　ログインに成功した場合

ンが成功した場合は Session から cookie のセット文を受け取り、そのセット文をスタートレスポンスに付随させる。また、その際は URL は/login のままで、HTML を書き換え、userpage に飛ぶためのハイパーリンクを載せたページを表示させる。図 5 はログインに失敗した場合の画面であり、図 6 はログインに成功した場合の画面である。このハイパーリンクを選択することで、userpage に遷移することができる。また、新規作成においても図 7 のとおり個別にエラー内容が存在している。

userpage

本ページではユーザの投稿した文章及びその時間の他、ユーザがフォロー及びフォロワーの人数とその表示行われるページへのリンク、timeline へのリンクが実装されている。/userpage でアクセスされた場合は cookie の token=でセットされた値から Session によってどのユーザがログインしているか受け取り、そのユーザの userpage を表示する。このページから各ユーザは投稿を行うことができる。/userpage?user=でアクセスされた場合そのクエリを受け取り、そのクエリの name のユーザの userpage を表示する。ただし、このページから投稿を行うことはできない。なおクエリ付きで userpage が読み込まれた場合そのユーザを Session によりログインされていると判断されたユーザがフォローをすることができ、またこのさいにフォローテーブルだけでなくフォロワーテーブルへの追加も同時に行う。/userpage?user=Session でログインし

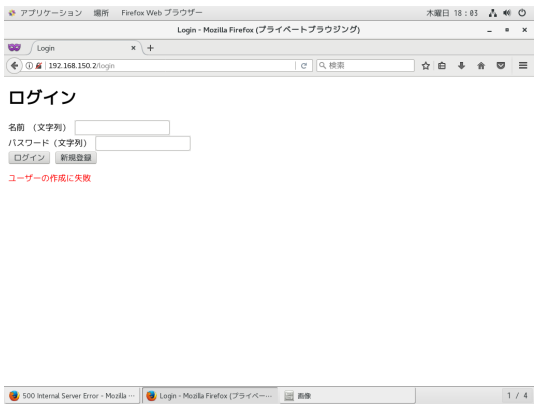

図 7　新規作成に失敗した場合

ている名前の場合は/userpage と同じページが出力されるようになっている。

userlist

本ページでは/timeline でのアクセスの場合は Session で獲得した user 名のフォローリスト及びフォロワーのユーザ名を表示する。この際、表示名にハイパーリンクをつけることでそのユーザの userpage に飛べるようになっている。/timeline?user=でアクセスされた場合そのクエリを受け取り、そのクエリの name のユーザの userlist を表示する。この際も、表示名にハイパーリンクをつけることでそのユーザの userpage に飛べるようになっている。また、userpage に戻るときは下のリンクを踏むことで実現する。その際クエリも同時に送信する。

timeline

本ページでは検索によって指定された条件を満たす投稿を全ユーザの投稿から獲得し、それを表示する。この際、何も条件が与えられない場合は全ユーザの投稿内容を日付順に表示する。

2.2 実験方法

本実験で使用した言語は Python でありその version は 3.6 であった。そのため、Python3 をインストールした後、以下のコマンドを入力して、必要な module 及びソフトウエアをインストールした。

```
yum install httpd-devel
yum install gcc
pip3.6 install mod_wsgi
yum install python36u-mod_wsgi
```

次に、wsgi の Apache ファイルを以下の内容にして/etc/httpd/conf.modules.d/10-wsgi-python3.6.conf に作成した。

```
# Only one mod\_wsgi can be loaded at a time.
# Don't attempt to load if already loaded.
<IfModule !wsgi_module>
    LoadModule wsgi_module modules/mod_wsgi_python3.6.so
    WSGIScriptAlias /login /var/www/cgi-bin/login.py
    WSGIScriptAlias /userlist /var/www/cgi-bin/follow.py
    WSGIScriptAlias /timeline /var/www/cgi-bin/timeline.py
    WSGIScriptAlias /userpage /var/www/cgi-bin/userpage.py
    WSGIScriptAlias /logout /var/www/cgi-bin/logout.py
</IfModule>
```

プログラムのソースコードがあるディレクトリ及び databese.db に以下のコマンドを実行して、書き込み権限をつけた。

```
chmod 777 database.db
chmod 777 cgi-bin
```

2.3 考察

本アプリケーションでは timeline や userpage にて投稿の表示数に制限を持たせていない。そのため、実験で使用する程度の規模であれば問題ないが、もっと大多数である場合投稿を表示させる数に何らかの制限をつけることが望ましいと考えられる。しかし、単に、表示数に制限を設けただけではユーザからの反感をかうと考えられる。そのため、例えば、このボタンを押せばもうプラス 10 件の投稿を閲覧できる。さらに押せばもう 10 件といった具合で表示数を増やす方法が考えられる。

3 調査課題: WEB アプリケーションの脆弱性

Web アプリケーションの脆弱性とは、Web アプリケーションの動作に関連したシステムやそのプログラムの不備を指す。Web アプリケーションの脆弱性が露呈すると、サイバー攻撃の被害を受ける可能性は高くなるため、脆弱性への対策は Web アプリケーションを立ち上げる上では必要不可欠である。代表的な脆弱性にはクロスサイトスクリプティング、SQL インジェクション、強制ブラウジングなどがある。

3.1 クロスサイトスクリプティング

クロスサイトスクリプティング (XSS) とは、ユーザからの入力内容を Web ページに表示する Web アプリケーションにおいて、ウェブサイト（標的サイト）の脆弱性 (XSS 脆弱性) を利用した攻撃手法を指す。具体的に言えば、例えば、掲示板などで投稿内容欄にブラウザ画面で処理可能なコマンドを含む文字列で罠画面内容を入力することで、リンクなどを作成し、これをほかのユーザが実行すると、セキュリティ的に問題のある別のウェブサイト（クロスサイト）に対し、脆弱性を利用した悪意を持った実行内容（スクリプト）が含まれた通信が実行される。この投稿される内容は具体的には HTML でできたドキュメントと、HTML に埋め込む形のスクリプト（JavaScript が一般的）である。サーバの管理者はこれを防ぐために入力できる型や値の制限や、サニタイジングによって、スクリプトが必要とする&や”といった文字列を置換することで、スクリプトを無害化をする必要がある。[2]

3.2 SQL インジェクション

SQL インジェクションは Web アプリケーションにおけるエスケープ処理が適切に行われていない脆弱性を狙った攻撃であり、インターネットの Web サイトなどの入力画面に対して、直接 SQL 命令文の文字列を入力することで、データベースに不正アクセスを行い、情報の入手や、データベースの破壊、Web ページの改ざんなどを行うことを指す。SQL インジェクションへの対策を行っていない Web サイトでは、例えばログイン画面でパスワードの欄に不正なデータベース命令を実行するための文字列を入力すると、パスワードを知らない攻撃者が正当な利用者としてログインし、個人情報を窃取したりすることできる。サーバの管理者はこれを防ぐために、不正な入力値による処理を防ぐことや、システムから表示されるエラーメッセージをそのまま表示しないようして、攻撃者に対してヒントを与えてしまうことを防ぐこと、システムで利用するデータベース

アカウントに対しては、最小限の権限だけを設定することなどをする必要がある。[3]

3.3 強制ブラウジング

強制ブラウジングとは、Web ページ上からリンクを辿ることをせず、アドレスバーから URL を直接入力することで、本来システム側では公開していないはずのディレクトリやファイルなどにアクセスを試みて、強制的にブラウザに表示させる攻撃である。強制ブラウジングにはいくつか種類があり、それにより、アクセス方法も異なる。ディレクトリ・リスティングはそのうちの一つであり、アドレスバーに

```
http://www.example.co.jp/usr/
```

という URL を入力したとする。これはルート・ディレクトリ配下の usr/というフォルダへのアクセスを意味するのだが、Web サーバ側の設定によっては、usr/フォルダ内のすべてのファイルがリスト表示されてしまう。これはディレクトリ・インデックスという標準搭載された機能が有効になっている状態で、上述のようなディレクトリへのアクセスが行われた場合に起こるものである。また、

```
http://www.example.co.jp/some/cgi-bin/filename2.html
```

という URL があった時、攻撃者は

```
http://www.example.co.jp/some/cgi-bin/filename1.html
http://www.example.co.jp/some/cgi-bin/test
http://www.example.co.jp/some/cgi-bin/
```

といったような様々な URL を想像することができ、Web サイト上でどのページからもリンクされていなくても、さらに実在するかどうかにも関係なく、攻撃者は思いつく限りのパターンでアクセスを試みることが可能になってしまう。この際ファイルに正しくアクセス権が設定されていなければ、結果として、機密情報ファイルに直接アクセスされる恐れがある。[4]

3.4 実験で作成したアプリケーションの安全性

本実験で作成したアプリケーションでは、ユーザの通信は http で行われ、暗号化された通信路を使用していないため通信の追跡が行われやすく、セキュリティという面でいえば非常に危険性の高いものである。また、login および投稿の際の SQL インジェクションやクロスサイトスクリプティング対策は何一つ行っていないため、html のスクリプト文や、SQL の実行文が入力された際には本アプリケーションでは太刀打ちできないものであると考えられる。
一方でデータベースのほうには多少のセキュリティ対策がされており、login テーブルのパスワードはハッシュ化されて保存されているため、パスワードの漏洩が起こってもパスワードの直ちの特定とはいかないようになっている。

4　まとめ

本レポートでは、WWW サーバ上で SQL データベースと協調することで動作する WEB アプリケーションをグループで作成を行った。SQL のデータベースに対する処理方法を理解した。また、グループ開発であったため皆がアクセスするデータベースへの処理を抽象化することによってコードの可読性が上昇するとともに、グループ間での処理の相違の防止が行えることや、プログラムの書きやすさが上昇することがわかった。

参考文献

[1] [SQL] データベース — TECHSCORE(テックスコア)
https://www.techscore.com/tech/sql/

[2] クロスサイトスクリプティング（XSS）のセキュリティ対策とは？ — セキュリティ対策 — CyberSecurityTIMESHttps://www.shadan-kun.com/blog/measure/1052/

[3] SQL インジェクションへの対策｜情報管理担当者の情報セキュリティ対策｜企業・組織の対策｜国民のための情報セキュリティサイト http://www.soumu.go.jp/main_sosiki/joho_tsusin/security/business/admin/06.html

[4] アプリケーションの攻撃パターン - 3) 強制ブラウズ — AppScan — テクマトリックス株式会社 http://www.techmatrix.co.jp/product/appscan/w_attackofappli/attack3_browse.html