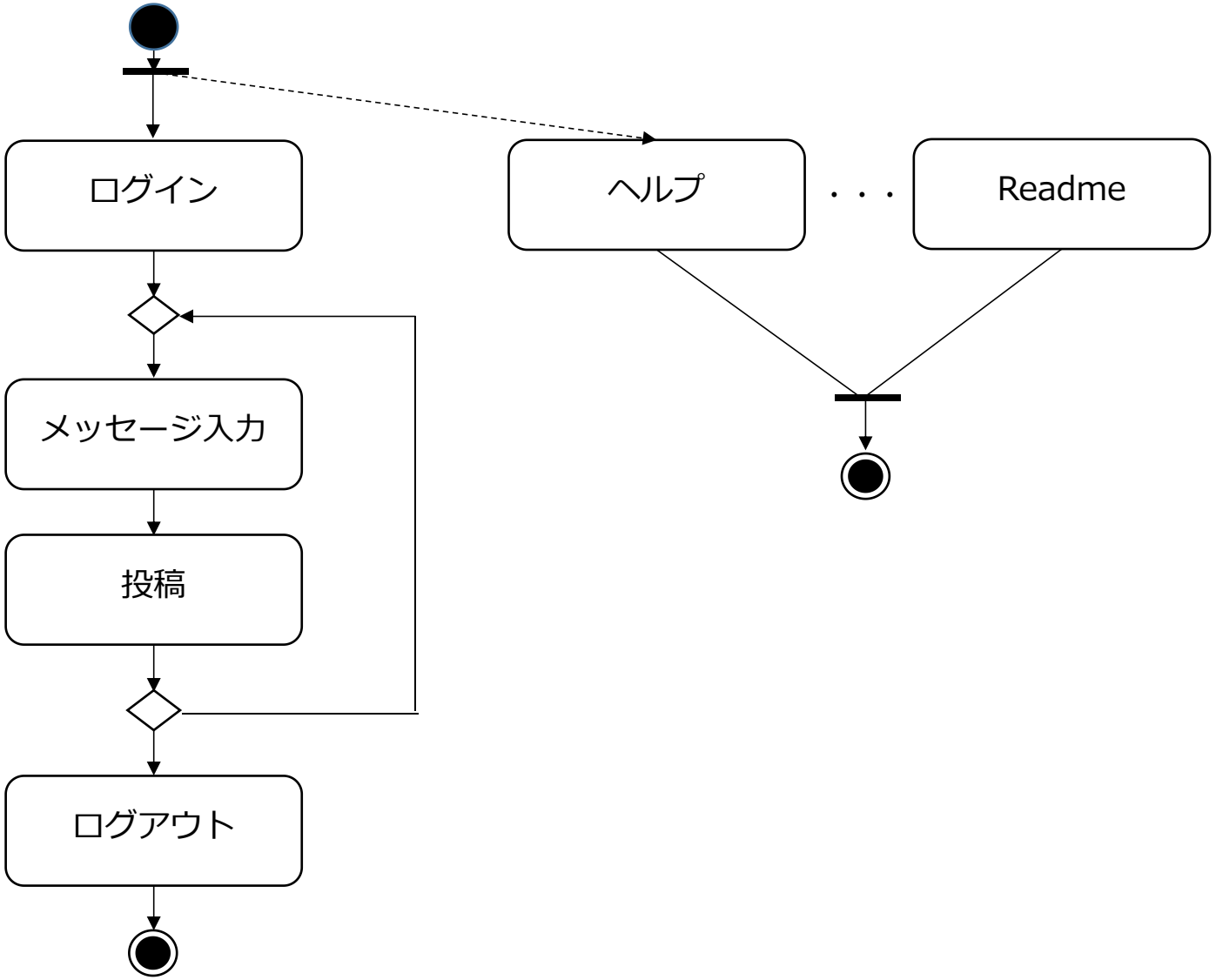
Sinatra web アプリ

toy-chat project

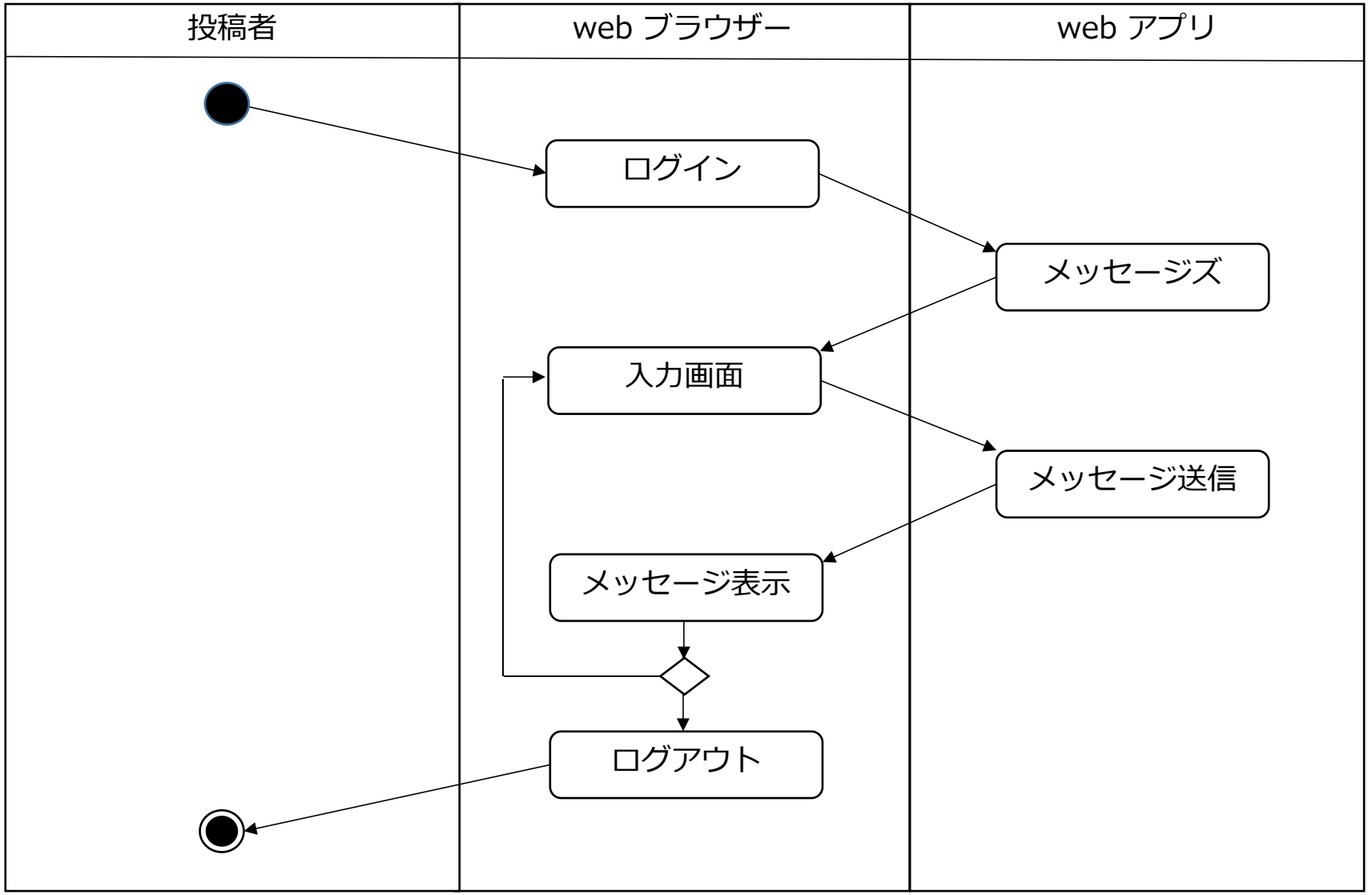
Sinatra webアプリ：toy-chatのシナリオ

- アプリの初期画面は「Welcome」であり、内容は特に規定しないが、以下のようなページリンクを設ける
  - ログインページ (必須)
  - ヘルプ、Readme など (オプション)
- ログインページは「チャット」を参加する人の名前を取得するページであり、取得した「情報」はサーバに保持しない。Webクライアント側(cookies)に保持するか、ページに保持するか、厳密に規定しない
- ログインページに参加者情報(名前)を取得し、チャットページへ遷移する
- チャットページには以下のフィールドが表示される
  - 参加者情報：ログイン者名が必須、そのたログイン時間などはオプション
  - 会話入力欄：「メッセージ」を入力した後、[送信]ボタンを押し、サーバにメッセージを送信する。なお、メッセージは128文字以内の制限を設けるが、制限を超えた文字の処理は実装に任せる（切り捨てか、エラーを返すか･･･）
  - 会話記録：以前から入力した「メッセージ」を時系列の逆順(最新情報は上)で表示され、メッセージは以下のフォーマットとなっている。
    - 投稿者名(ログイン時の名前)、投稿時間
    - メッセージ（128文字以内）
    - 空白行 （メッセージの分離は「横線」でもOK）

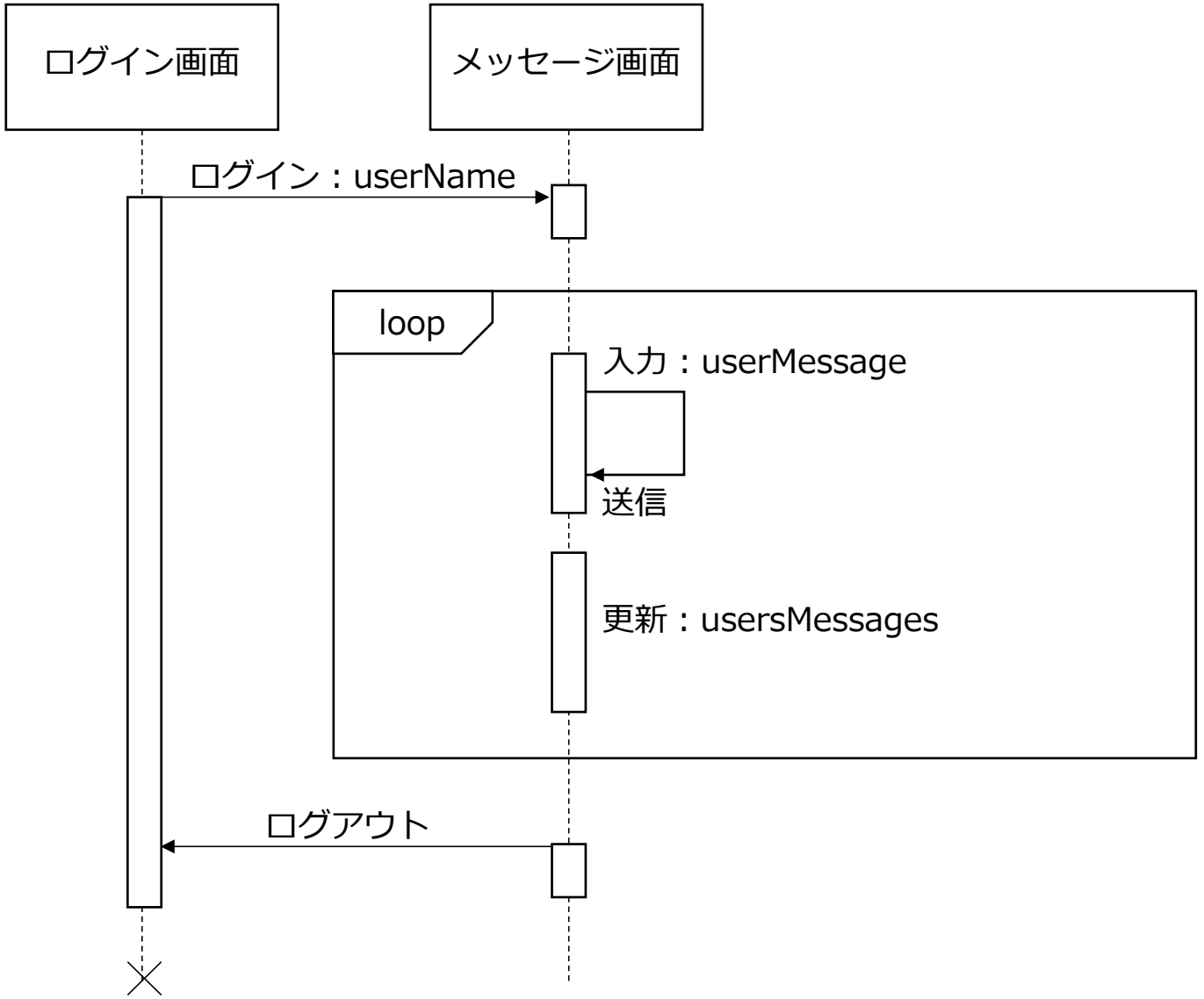
Toy-chatアクティビティ図(1)



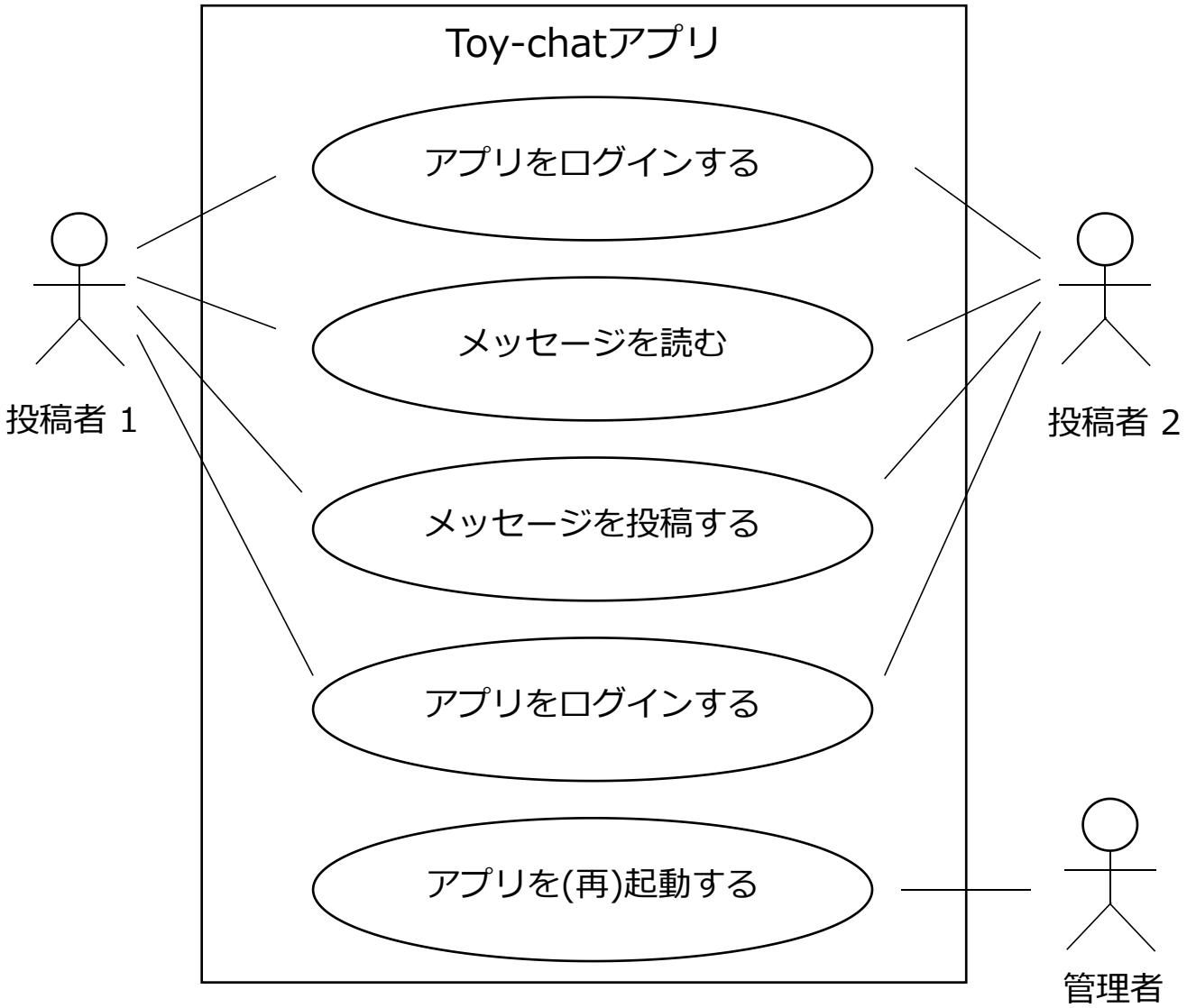
Toy-chatアクティビティ図(2)



Toy-chatシーケンス図



Toy-chatユースケース



検証(サンプル)例：

参考資料

/main.rb

```
require ‘Sinatra’

get ‘/’ do
    @hello = “こんにちは、”
    erb  :index
end

post ‘/form’ do
    @input1 = params[:input1]
    @hello = @input1
    erb :index
end
```

/views/index.erb

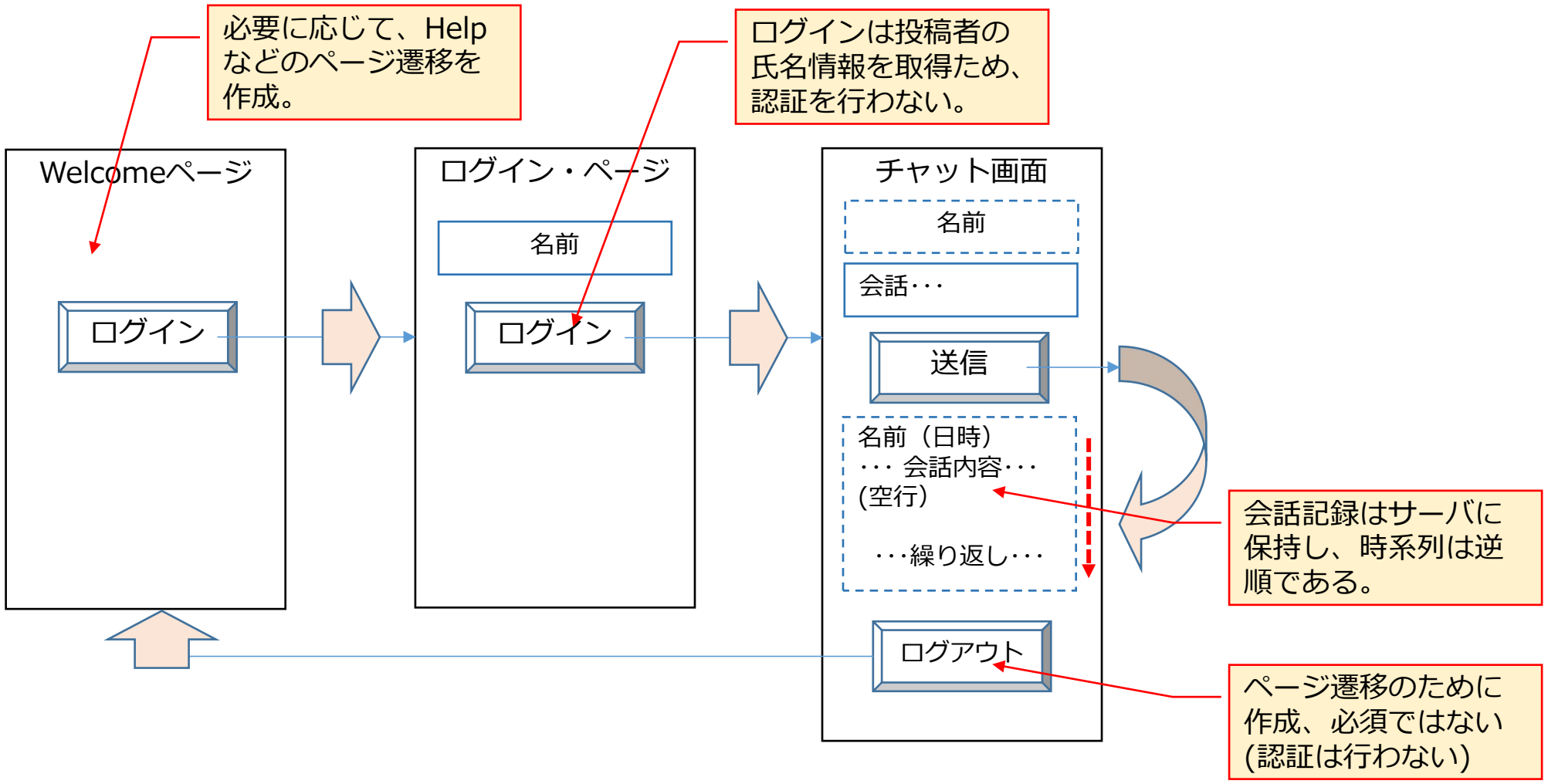
```
<ul>
    <li>ここにエントリが表示される</li>
    <li><%= @hello %></li>
</ul>
<ul>
    <form method=“POST” action=“/form”>
        <input type=“test” name=“input1” valude=“”>
        <input type=“submit” valude=“送信”>
    </form>
</ul>
```

/views/layout.erb

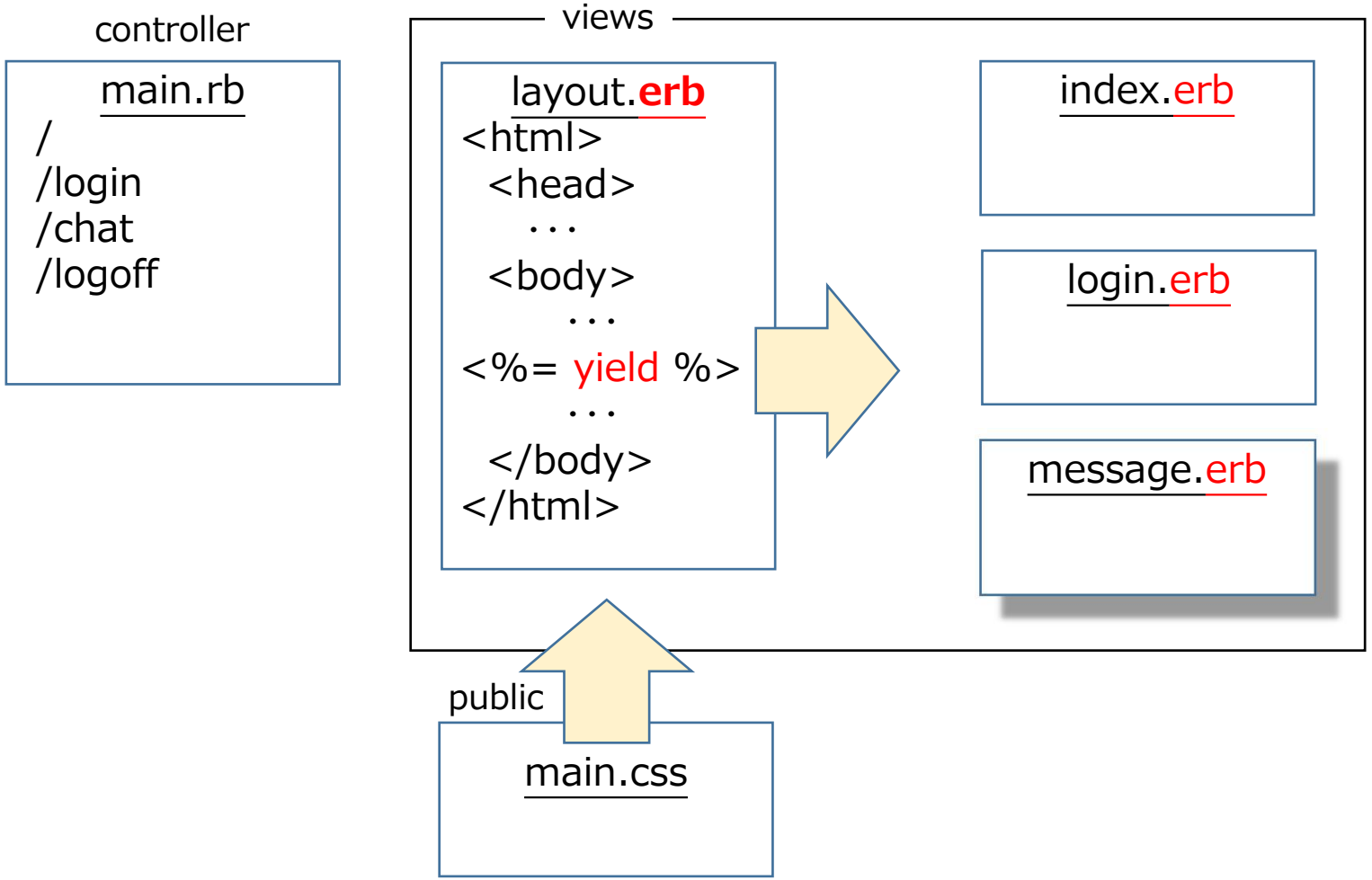
```
<!DOCTYPE html>
<html lang=“ja”>
    <head>
        <meta charaset=“utf-8”>
        <title>サンプル</title>
    </head>
    <body>
        <h1>*** チャット・プログラム ***</h1>
        <%= yield %>

    </body>
</html>
```

アプリの状態遷移図


必要に応じて、Help
などのページ遷移を
作成。
ログインは投稿者の
氏名情報を取得ため、
認証を行わない。
Welcomeページ
ログイン
ログイン・ページ
名前
ログイン
チャット画面
名前
会話･･･
送信
名前（日時）
･･･ 会話内容･･･
(空行)
･･･繰り返し･･･
ログアウト
会話記録はサーバに
保持し、時系列は逆
順である。
ページ遷移のために
作成、必須ではない
(認証は行わない)

Top-chatアプリのファイル詳細



Top-chatアプリのファイル構成(3)

// プロジェクト構成： Github://a1852rw/shinatora_enpit
shinatra_enpit

- Gemfile
- config.ru // Rackの設定ファイル。最初に実行されるファイル
- main.rb // controller
- app/ // 何らかの処理用
  - module/ // モジュール類のファイル
- views/ // ビューのテンプレート（haml）ファイル
  - index.html
  - layout.erb // 基本テンプレートファイル
  - style/ // sass（scss）ファイル
    - baes.scss
- db/ // データベースの設定ファイル
- log/ // ログ置き場
- public/ // webサーバのドキュメントルートはここを指定、画像・CSSなどもここ
  - main.css // 共用 css ファイル
- tmp/
- vendor/
  - Bundle/ // gemの管理ディレクトリ
- unicorn.rb // unicornの設定ファイル

アプリのファイル構成(1)

参考資料

// 最小構成：
Projectname/

- main.rb　　// controller
- config.ru　　// Rackの設定ファイル。最初に実行されるファイル
- views/　　// ビューのテンプレート（haml）ファイル
- public/　　// webサーバのドキュメントルートはここを指定、画像・CSSなどもここ

アプリのファイル構成(2)

参考資料

// 参考構成： https://qiita.com/hiroki_y/items/06f5780543bec988d8b7
Projectname/

- Gemfile
- config.ru // Rackの設定ファイル。最初に実行されるファイル
- main.rb // controller
- app/ // 何らかの処理用
  - module/ // モジュール類のファイル
- views/ // ビューのテンプレート（haml）ファイル
  - index.haml
  - layout.haml // 基本テンプレートファイル
  - style/ // sass（scss）ファイル
    - baes.scss
- db/ // データベースの設定ファイル
- log/ // ログ置き場
- public/ // webサーバのドキュメントルートはここを指定、画像・CSSなどもここ
- tmp/
- vendor/
  - Bundle/ // gemの管理ディレクトリ
- unicorn.rb // unicornの設定ファイル