ChatChat.com

外部仕様書, CHOE CHUNGIL

最終更新日時 : 2019年9月2日13:00

目次

[1. はじめに 3](#_Toc19204294)

[2. シナリオ 3](#_Toc19204295)

[2.1 シナリオ1 : ハナ 3](#_Toc19204297)

[3. 対象外 4](#_Toc19204298)

[4. 概略フローチャート 4](#_Toc19204299)

[5. 画面ごとの仕様 5](#_Toc19204300)

[5.1 初期ページ(Index) 5](#_Toc19204304)

[5.2 会員登録ページ(Join) 6](#_Toc19204305)

[5.3 ホームページ(Main) 7](#_Toc19204306)

[5.4 Chat 9](#_Toc19204307)

[5.5 チャットルーム 10](#_Toc19204308)

[5.6 セットアップ 11](#_Toc19204309)

[5.7 ID削除 11](#_Toc19204310)

[5.8 Open Chat 機能 12](#_Toc19204311)

[6 開発詳細 13](#_Toc19204312)

[6.1 開発環境 13](#_Toc19204313)

[6.2 開発期間 13](#_Toc19204314)

[7 テストシナリオ 14](#_Toc19204315)

[7.1 サインアップ 14](#_Toc19204316)

[7.2 ログイン 14](#_Toc19204317)

[7.3 友達追加 14](#_Toc19204318)

[7.4 友達削除 15](#_Toc19204319)

[7.5 個人チャット 15](#_Toc19204320)

[7.6 グループチャット 16](#_Toc19204321)

[7.7 オープンチャット 17](#_Toc19204322)

[7.8 チャット 17](#_Toc19204323)

[7.9 ログアウト 18](#_Toc19204324)

[7.10 アカウント削除 18](#_Toc19204325)

# はじめに

ChatChat.comは、ウェブの上でチャットを行うサービスである。

# シナリオ



## シナリオ1 : ハナ

ハナはとある大学の学生で、とある国際学会にスタッフとして参加することになった。この国際学会は、世界各国の研究者とスタッフが参加する学会で、スタッフ同士に色んなコミュニケーションを行う必要があった。

ここで発生した問題は、メールを利用してコミュニケーションを行う場合には、学会中、一日数十、数百通のメールを交換しなければならない必要があり、メールボックスが混雑になる可能性があるということだった。また、従来のチャットアプリケーションの場合、国によって利用できないアプリが存在することだった。

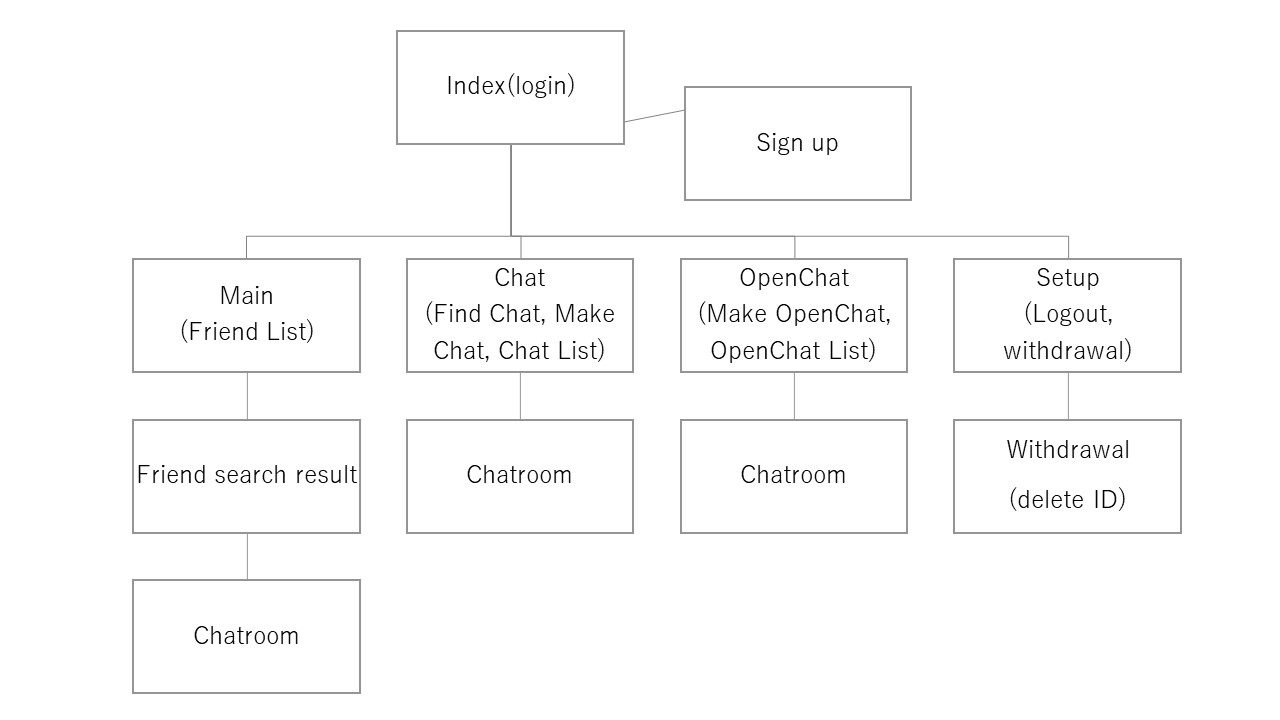
ハナは、これらの問題を解決するために、Web上で利用できる同時に、複数の人数の参加が簡単なチャットサービスCHATCHAT.COMを利用することにした。 CHATCHAT.COMは、世界のどの国でも接続が可能であり、チャットルームのIDを教えるだけで、簡単に使用することができるというメリットがあるアプリである。

ハナは、一日に何度もCHATCHAT.COMにアクセスし、スタッフグループチャットでコミュニケーションを行うことで学会を無事終えることができた。

# 対象外

今回実際のサーバー上で通信するシステムは作成しない。

# 概略フローチャート

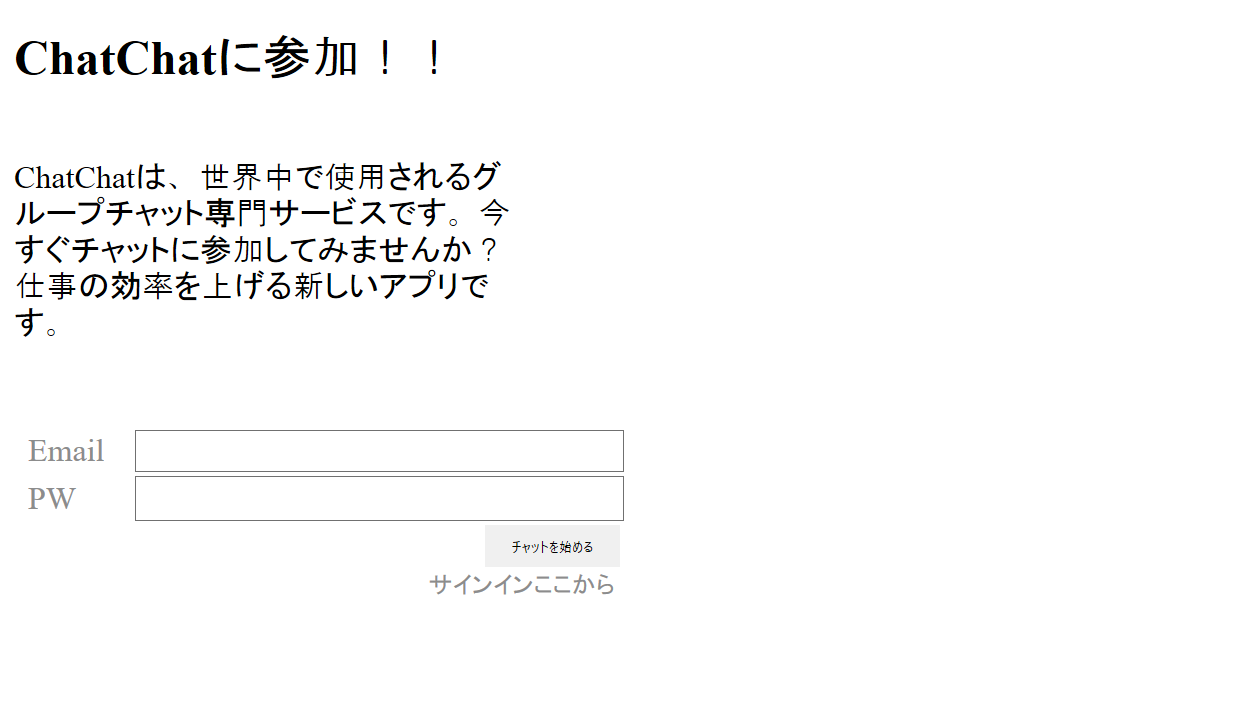


# 画面ごとの仕様



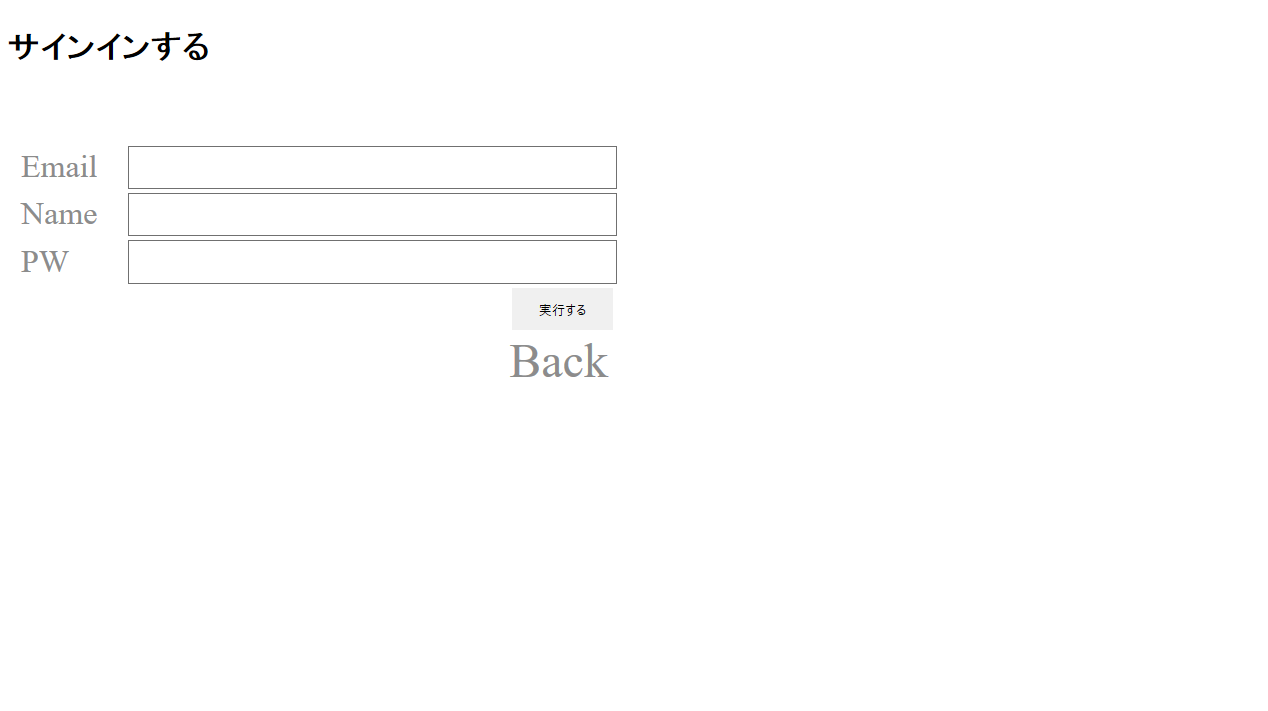
## 初期ページ(Index)

最初のページで、ログイン、会員登録ページに移動するリンクが存在する。



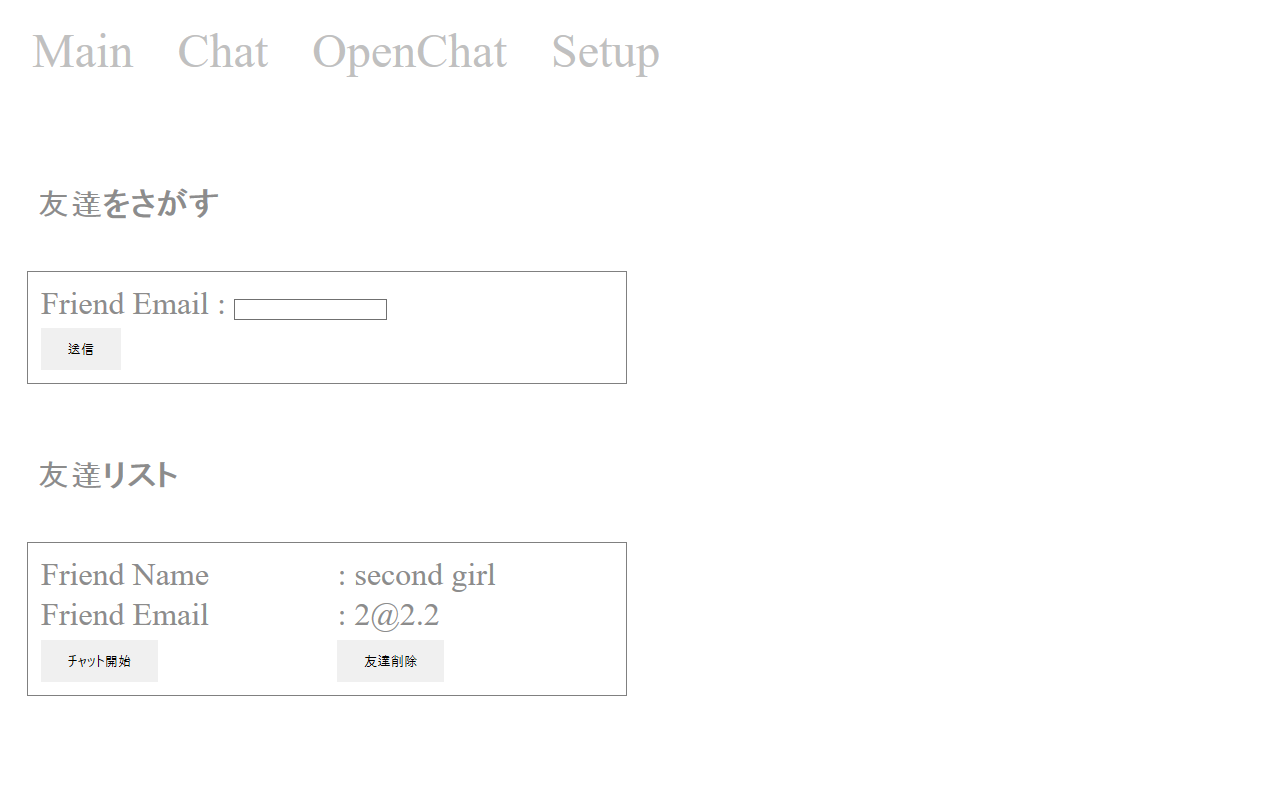
## 会員登録ページ(Join)

会員登録のページで、名前、電子メール、パスワードを入力して会員登録を行うことができる。

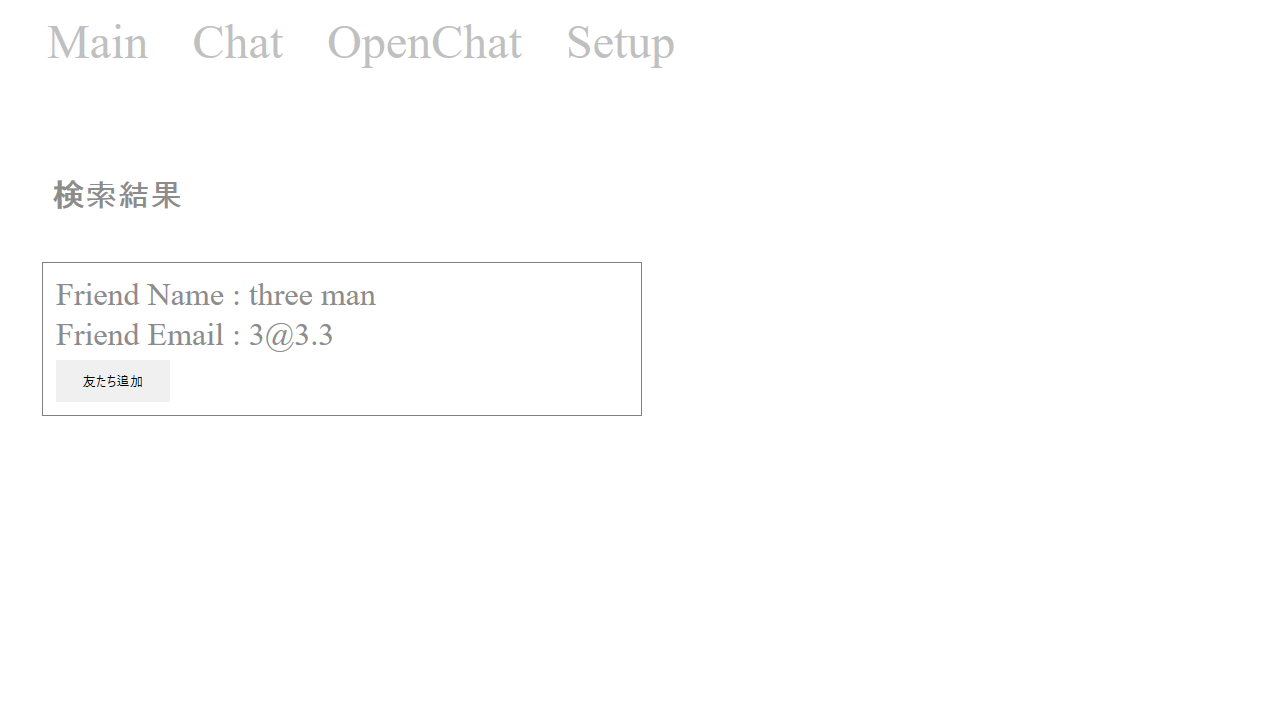


## ホームページ(Main)

基本的にはフレンドリストがデフォルト設定になっていて、メールアドレスで友達を見つけるFind Friend機能と現在の友達リストを見せる機能がある。

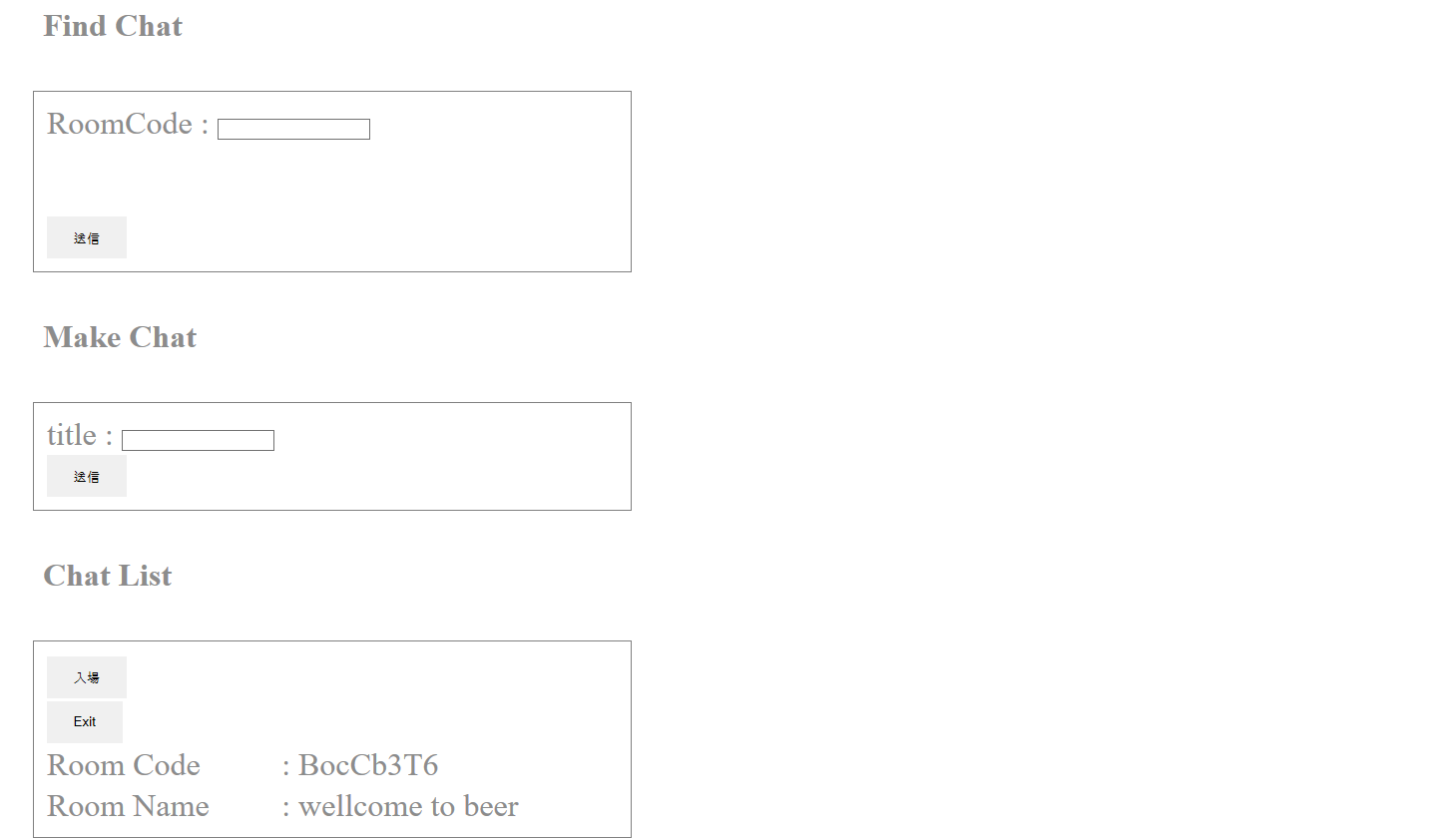


Friendの情報は、メールアドレス、名前が表示され、チャットの参加(作成)、友人を削除する機能がある。メールアドレスを入力して友達を見つけることができる。下の画像は、電子メールに対応する友達を発見した場合のイメージである。メールアドレスが存在しない場合は、Mainページから変化しない。



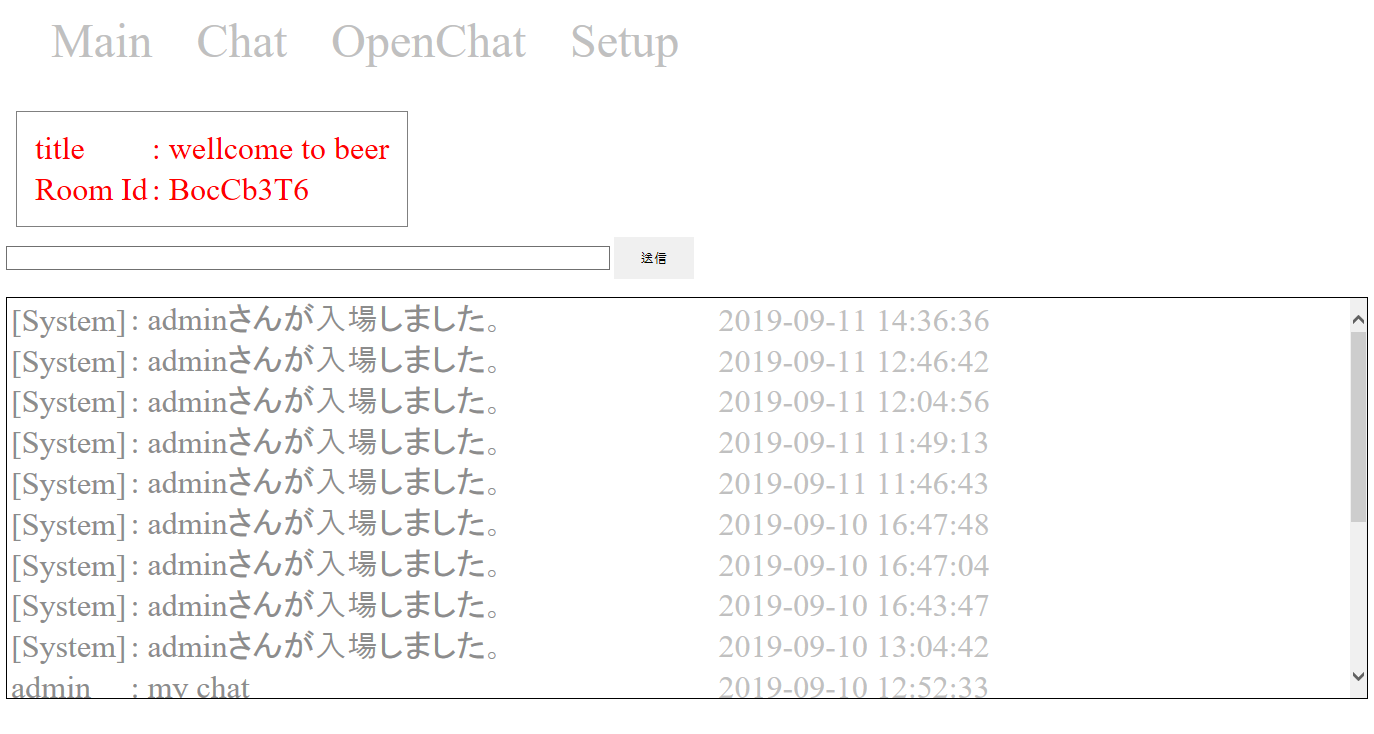
## Chat

チャット(Chat)のデフォルト設定は、チャットを探す機能、新しいチャットの作成、チャットリストの表示で構成されている。チャットルームの名前を付けて作成すると、自動的にチャットルームが作られる。作成されたチャットルームは、ルームコードを持つことになり、コードを他人に渡してチャットに参加できるようにする。



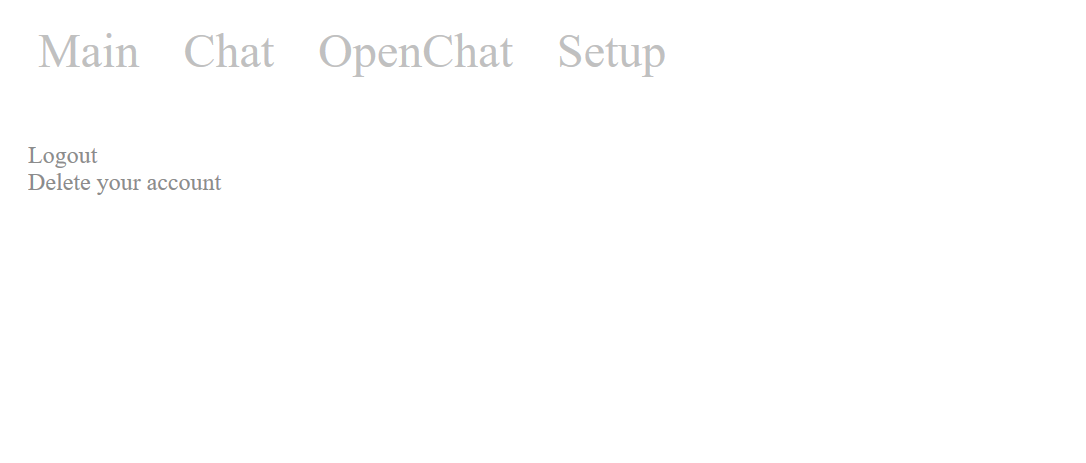
## チャットルーム

下の画像がチャットルーム内の様子である。チャットルームの名前とルームコードが表示され、下にはチャットの内容が表示されるウィンドウと、チャットを入力する入力ウィンドウ、入力ボタンで構成されている。チャットの内容は、[名前：内容、日時]で表示されている。



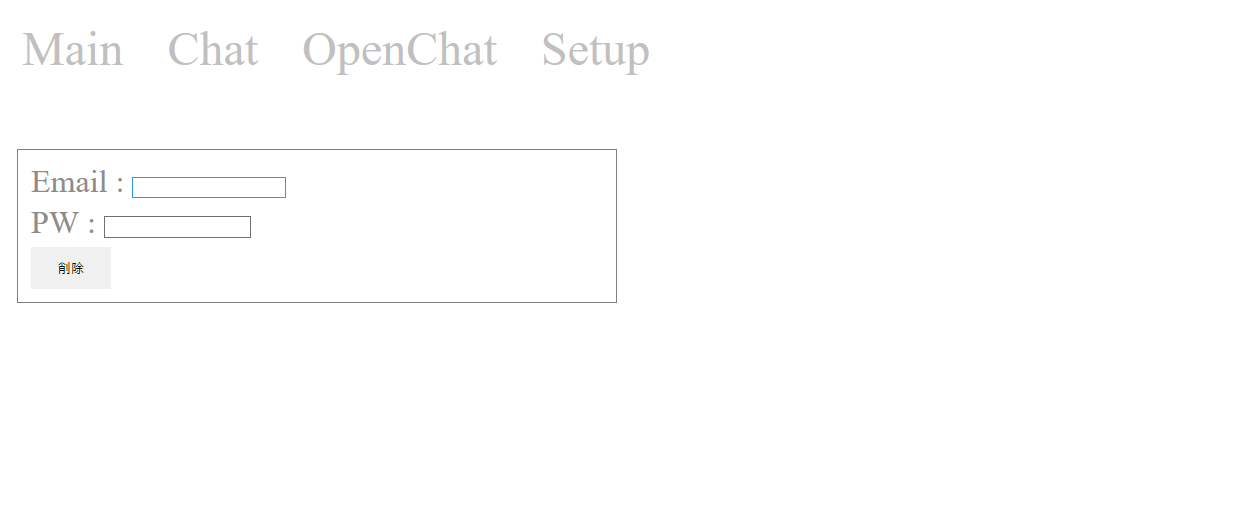
## セットアップ

セットアップページには、ログアウトする機能と、IDを削除する機能が存在する。



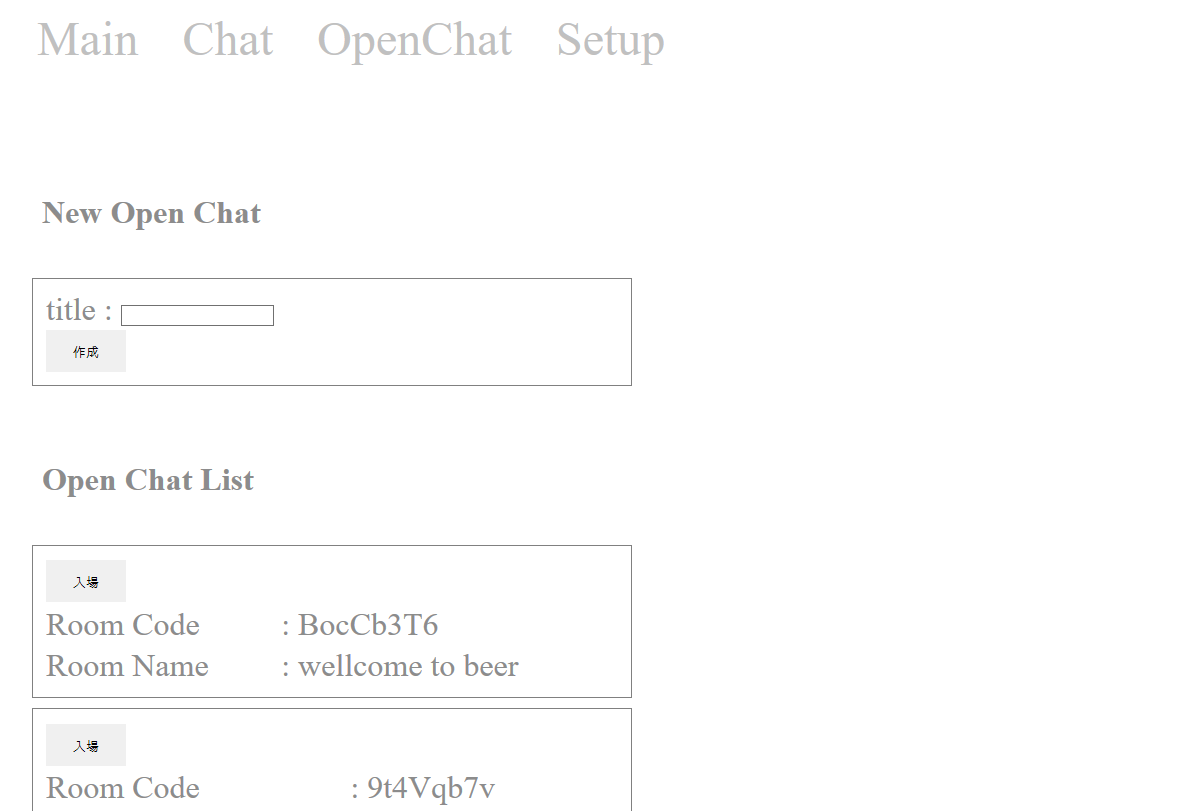
## ID削除

ID削除の場合、電子メールとパスワードを入力して削除するようになっている。現在のページでは、[I think more...]を入力すると削除を行わない。 [Delete Id]を入力すると、ユーザIDが削除されるように構成されている。



## Open Chat 機能

誰でも参加可能なチャット機能「Open Chat」。全体のOpen Chatリストから参加したいチャットを選択し、チャットメンバーになるシステムである。機能としては、新しいOpen Chatを作成する機能と、作成されているOpen Chatのリストを表示する機能で構成されている。



# 開発詳細

## 開発環境

開発言語：Servlet/Jsp (Java(v11), Tomcat(v9.0))

DBMS：MySQL (v8.0.17)

## 開発期間

2019年9月2日　14：00～2019年9月12日　18：00、9日間(68時間)

# テストシナリオ

## サインアップ

* Index.jspからスタート。「サインアップはここから」をクリックするとSignUp.jspに移動。
* Email、Name、 Password(PW)を入力するとSignUpServlet.javaを実行する。
* SignUpServlet.javaで登録されているEmail、Nameと、入力したEmail、Nameを比較して、両方重複してなかったらDB(webchat.user\_tb)に登録する。
* Nameに空欄が存在する場合、自動的にアンダーバー(\_)に変換される。
* 登録が完了したらIndex.jspに戻る。
* 重複してたらDBに接続せずにIndex.jspに戻る。

## ログイン

* Index.jspにあるログインフォームにEmail、Password(PW)を入力すると、LoginServlet.javaを実行する。(ログイン)
* FriendListServlet.javaを実行して友達の情報をDB(webchat.friend\_tb)から読んでから、Main.jspに移動して友達のリストを表示する。
* もし友達が存在しない場合にはEmptyFriendMain.jspに移動して新しい友達を検索するフォームだけを表示する。

## 友達追加

* Main.jsp又はEmptyFriendMain.jspの「友達をさがす」でEmail入力し送信をクリックすると、そのEmailの友達の「名前」と「メールアドレス」がFriendSearchResult.jspに表示される。
* 「友達追加」をクリックするとAddFriendServlet.javaを実行して友達になる。友達の関係情報はDB(webchat.friend\_tb)に登録する。
* ただし、友達は一方的に追加するだけで、お互い友達になるためには、お互い友達追加を実行する必要がある。

## 友達削除

* Main.jspの「友達リスト」から、削除するユーザーのフォームから「友達削除」をクリックすると、DeleteFriendServlet.javaが実行される。
* DeleteFriendServlet.javaでは対象になるユーザーの情報を友達登録したDBから削除する。
* FriendListServlet.javaを実行し、友達リストを更新してMain.jspに移動する。

## 個人チャット

* 個人チャットはMain.jspの友達リストから作成、参加である。
* 個人チャットはメニューの「Chat」(Chat.jsp)のチャットリストからも確認、参加可能である。
* Main.jspの友達リストから「チャット開始」をクリックすると、FriendChatServlet.javaを実行する。
* １．友達と自分の個人チャットが存在する場合、EnterRoomServlet.javaを実行してChatRoom.jspに移動する。
* 個人チャットルームの名前は「個人チャット」に固定される。
* ２．個人チャットが存在しない場合、新しい個人チャットを作成して、自分と友達をそのチャットに参加させる。個人チャットで使用するDBは下のようになっている。
* webchat.chatroom\_tb : 全てのチャットルームを登録するデータベース
* webchat.smallchat\_tb : 個人チャットと、そのチャットに参加している人を登録するデータベース
* webchat.join\_tb : 全てのチャットルームを対象とする、チャットルームとそのチャットに参加している人を登録するデータベース

## グループチャット

* グループチャットはメニューの「Chat」から作成、参加が可能である。
* メニューから「Chat」をクリックすると、ChatListServlet.javaが実行される。
* ChatListServlet.javaでユーザーが参加しているチャットルームの情報をDBから読んで、Chat.jspに移動する。
* Chat.jspは「チャットを探す」, 「新しいチャットを始める」, 「チャットリスト」で構成されている。
* 「チャットを探す」でグループ名を作成し送信をクリックすると、MakeChatServlet.javaを実行する。
* MakeChatServlet.javaでは入力したグループ名と、ランダムに作成したルームコードをDBに登録して、EnterRoomServlet.javaを実行してChatRoom.jspに移動する。
* グループチャットはルームコードを入力することで誰でも参加が可能である。
* Chat.jspの「チャットリスト」には現在ユーザーが参加しているチャットルームがすべて表示される。

## オープンチャット

* オープンチャットはメニューの「OpenChat」から作成、参加が可能である。
* メニューから「OpenChat」をクリックすると、OpenChatListServlet.javaが実行される。
* OpenChatListServlet.javaではすべてのオープンチャットのリストをDBから読んで、OpenChat.jspに移動する。
* OpenChat.jspでは、新しいオープンチャットを作成する「新しいオープンチャットを始める」と、すべてのオープンチャットのリスト「オープンチャットリスト」が表示される。
* 「新しいオープンチャットを始める」で「オープンチャット名」を自由に作成し送信ボタンをクリックするとNewOpenChatServlet.javaを実行する。
* NewOpenChatServlet.javaでは入力した「オープンチャット名」と、ランダムに作成したルームコードをDBに登録した後、EnterRoomServlet.javaを実行してChatRoom.jspに移動する。
* OpenChat.jspのオープンチャットリスト」にはすべてのオープンチャットが表示されて、誰でも参加が可能である。

## チャット

* すべてのチャットはChatRoom.jspで行われる。
* ChatRoom.jspはEnterRoomServlet.javaを実行することで接続可能である。
* チャット名とルームコードが表示される。
* 参加ユーザーが表示される。
* メッセージを書いて「送信」をクリックすると、下に最新のメッセージが一番上に表示される。
* メッセージはユーザー名、作成日時、メッセージ内容が表示される。

## ログアウト

* メニューの「Setup」をクリックするとSetup.jspに移動する。
* Setup.jspの「ログアウト」をクリックするとLogoutServlet.javaが実行して、ログイン状態が解除される。（ログアウト）
* その後、自動的にIndex.jspに移動する。

## アカウント削除

* Setup(Setup.jsp)の「IDを削除する」をクリックするとWithdrawal.jspに移動する。
* Withdrawal.jspの「本当にIDを削除しますか？」の下のフォームにEmailとPassword(PW)を入力すると、WithdrawalServlet.javaを実行してDBからユーザーの情報を削除する。
* その後、自動的にIndex.jspに移動する。
* アカウントを削除すると、友達関係も削除される。
* アカウントを削除しても、チャットルームとメッセージは削除されない。