**Document out of date. Not fully updated.**

**These rules apply to all functions**

If requirements are not met, return:

Result: FALSE

Error: Requirements not met

Output: NULL

If session is expired, return:

Result: FALSE

Error: Session expired

Output: NULL

If a SQL statement is invalid, return:

Result: FALSE

Error: Server broke

Output: NULL

**User Action** – handles all actions related to users via the User object. **All Methods require $\_GET["action"]**

***Login***

**Requirement(s)**: $\_POST['user\_name'], $\_POST['password']

**Action**: Logs user into his account. Sets $\_SESSION['iUserId'] to user’s UserId

**Return**:

Result: TRUE/FALSE

Error: Incorrect Password or Username

Output: User ID

***Logout***

**Requirement(s)**: Nothing

**Action**: Unsets $\_SESSION['iUserId']. Logs user out.

**Return**:

Result: TRUE

Error: NULL

Output: NULL

***CreateUser***

**Requirement(s)**: $\_POST['user\_name'], $\_POST['password']

**Action**: Creates User object / account for user. Sets $\_SESSION['iUserId'] to user’s UserId

**Return**:

Result: True/FALSE

Error: Username is taken/ Alphanumeric characters only/ Password must be at least 5 characters

Output: User ID

***EditProfile***

**Requirement(s)**: $\_SESSION['iUserId'], $\_POST['name'], $\_POST['gender'], $\_POST['age'], $\_POST['blurb']

**Action**: Edits User object / account of user.

**Return**:

Result: TRUE

Error: NULL

Output: NULL

***ViewProfile***

**Requirement(s)**:$\_SESSION['iUserId']

**Action**: Displays all information about a User with $\_GET["iUserId"] (name, profile\_pic, is\_friend)

**Return**:

Result: TRUE/FALSE

Error: Account not found

Output: NULL

***EnableMatchMaking***

**Requirement(s)**:$\_SESSION['iUserId']

**Action**: Enables matchmaking for the current user. The user must have

**Return**:

Result: TRUE/FALSE

Error: Account not found

Output: NULL

**Meal Action** – handles all actions related to meals via the Meal object. **All Methods require $\_SESSION['iUserId'] and $\_GET["action"]**

***InviteGuest***

**Requirement(s)**: $\_POST["InvitedUserNames"], $\_POST["token"]

**Action**: Invites users to meal with token $\_POST["token"]. Sets invited User states to INVITED

**Return**:

Result: TRUE/FALSE

Error: Meal not found, Account not found

Output: NULL

***AcceptInvite***

**Requirement(s)**: $\_GET["token"]

**Action**: Accept invite to meal with token $\_GET[“token”]. Sets User state to GUEST. Sets time sheet to active time sheet.

**Return**

Result: TRUE/FALSE

Error: Meal was either cancelled or completed / Invite not found / No active time sheet

Output: NULL

***RejectInvite***

**Requirement(s)**: $\_GET["token"]

**Action**: Reject invite to meal with token $\_GET[“token”]

**Return**:

Result: TRUE

Error: Invite not found

Output: NULL

***ViewMealUsers***

**Requirement(s)**: $\_GET["token"].

**Action**: Displays information about the attendees of a Meal (user\_name, state, prof\_img)

**Return**:

Result: TRUE/FALSE

Error: Meal doesn’t exist

Output: user\_name, state, prof\_img

***ViewMealState***

**Requirement(s)**: $\_GET["token"].

**Action**: Displays information about a Meal (meal state, time meal is happening, restaurant meal is happening in, restaurant location)

**Return**:

Result: TRUE/FALSE

Error: Meal doesn’t exist

Output: meal\_state, meal\_time\_slot, restaurant\_name, restaurant\_location

***ViewMealTimes***

**Requirement(s)**: $\_GET["token"].

**Action**: Displays information about the times of a Meal (start\_time, end\_time, day) along with information about each user who can attend at that time (user\_name, state, prof\_img)

**Return**:

Result: TRUE/FALSE

Error: Meal doesn’t exist

Output: start\_time, end\_time, day, user\_name, state, prof\_img

***LeaveMeal***

**Requirement(s)**: $\_GET["token"]

**Action**: Leave meal with token $\_GET[“token”]. If HOST leaves, meal is set to CANCELLED

**Return**

Result: TRUE / FALSE

Error: Meal was either cancelled or completed / Meal not found / Meal user not found

Output: NULL

***ListMeals***

**Requirement(s)**: Nothing

**Action**: List all meals that the user is a part of and invited to. Displays information about all the meals as well (meal state, token, meal name, user id, user state, user pic)

**Return**:

Result: TRUE

Error: NULL

Output: NULL

***CreateMeal***

**Requirement(s)**:Nothing. Optional $\_POST["meal\_name"],

**Action**: Creates meal. Sets the Meal state to TIME\_VOTING. Sets the User state to HOST.

**Return**: Nothing

Result: TRUE

Error: No active time sheet

Output: Meal token

***SelectTime***

**Requirement(s)**: $\_POST["token"], $\_POST["time\_slot"]

**Action**: Sets the time of a meal. Sets the Meal state to REST\_VOTING.

**Return**:

Result: TRUE

Error: Meal not found / Meal user not found / Not the host

Output: NULL

***SelectRestaurant***

**Requirement(s)**: $\_POST["token"], $\_POST["restaurant\_name"], $\_POST["restaurant\_location"]

**Action**: Sets the restaurant name and restaurant location of a meal. Sets the Meal state to COMPLETED

**Return**

Result: TRUE

Error: Meal not found / Meal user not found / Not the host

Output: NULL

***CompleteMeal***

**Requirement(s)**: $\_GET["token"]

**Action**: Sets the meal state to COMPLETED. Only the meal host can do this.

**Return**

Result: TRUE

Error: Meal not found / Meal user not found / Not the host

Output: NULL

***ChangeMealName***

**Requirement(s)**: $\_POST["token"], $\_POST["meal\_name]

**Action**: Sets the meal name. Only the meal host can do this.

**Return**

Result: TRUE

Error: Meal not found / Meal user not found / Not the host

Output: NULL

**Restaurant Action** – handles all actions related to votes for a meal’s restaurant via the MealRestaurant object. **All Methods require $\_SESSION['iUserId'] and $\_GET["action"]**

***ListVotes***

**Requirement(s)**: $\_GET["token"]

**Action**: List all restaurant votes for meal with token $\_GET["token"]. Displays information about votes (user name, prof pic, restaurant id, restaurant name, state, id)

**Return**

Result: TRUE/FALSE

Error: Meal not found

Output: Displays information about votes (user name, prof pic, restaurant id, restaurant name, state, id)

***CreateVote***

**Requirement(s)**: $\_POST["token"], $\_POST["restaurant\_id"], $\_POST["state"], $\_POST[“restaurant\_name”]

**Action**: Creates restaurant vote for meal with token $\_GET[“token”]

**Return**

Result: TRUE/FALSE

Error: Meal not found / Already voted for this Restaurant

Output: NULL

***DeleteVote***

**Requirement(s)**: $\_GET["mr\_id"]  
**Action**: Deletes restaurant vote with id $\_GET["mr\_id"]

**Return**:

Result: TRUE/FALSE

Error: Vote not found

Output: NULL

**Relation Action** – handles all actions related to user relationships via the UserRelation object. **All Methods require $\_SESSION['iUserId'] and $\_GET["action"]**

***ListRelations***

**Requirement(s)**: Nothing

**Action**: List all relations that the user is in. Displays information about relations (user name, first name, last name, state)

**Return**:

Result: TRUE/False

Error: Null

Output: Displays information about relations (user name, user\_id, state, user img)

***CreateRelation***

**Requirement(s)**: $\_GET[‘user\_name']

**Action**: Creates UserRelation between currently logged in user and $\_GET[‘user\_name']. Sets the state to INVITED.

**Return**:

Result: TRUE/False

Error: Can’t friend yourself / Relation already exists / Accout not Found

Output: NULL

***AcceptRelation***

**Requirement(s)**: $\_GET["user\_name"]

**Action**: Accept a friendship request from user with $\_GET["user\_name"]. Sets the UserRelation state to ACCEPTED.

**Return**:

Result: TRUE/False

Error: Relation not found / Account not found

Output: NULL

***DeleteRelation***

**Requirement(s)**: $\_GET["user\_name"]  
**Action**: Deletes user relation with $\_GET["user\_name"]

**Return**: Nothing

Result: TRUE

Error: Account not found

Output: NULL

**Time Action** – handles all actions related to user time slots via the TimeSlot object. **All Methods require $\_SESSION['iUserId'] and $\_GET["action"]**

***ListTimes***

**Requirement(s)**: $\_GET['sheet\_id'])

**Action**: List all time slots of user. Displays information about the user (time slot)

**Return**:

Result: TRUE

Error: Timesheet not found

Output: : List all time slots of user. Displays information about the user (time slot)

***CreateTime***

**Requirement(s)**: $\_POST['start\_time'], $\_POST['end\_time'], $\_POST['day'], $\_POST['sheet\_id']. start\_time and end\_time have the following format: XX:XX for example 23:30. day can be MON, TUE, WED, THU, FRI, SAT, SUN

**Action**: Creates time slot for the logged in user.

**Return**

Result: TRUE/False

Error: Timeslot invalid / Timeslot overlap / Timesheet not found

Output: NULL

***DeleteTime***

**Requirement(s)**: $\_POST["time\_slot"], $\_POST["sheet\_id"].   
**Action**: Deletes time slot with $\_POST["time\_slot"] & $\_POST["sheet\_id"].

**Return**:

Result: TRUE

Error: Timesheet not found

Output: NULL

***ListTimeSheets***

**Requirement(s)**: None

**Action**: List all time sheets of user. Displays information about the timesheets (sheet id, name, state)

**Return**:

Result: TRUE

Error: NULL

Output: List all time sheets of user. Displays information about the timesheets (sheet id, name, state)

***CreateTimeSheet***

**Requirement(s)**: $\_POST[‘name’]

**Action**: Creates time sheet for the logged in user. Sets the State to Inactive

**Return**

Result: TRUE/False

Error: Timesheet exists

Output: NULL

***DeleteTimeSheet***

**Requirement(s)**: $\_GET ["sheet\_id"]  
**Action**: Deletes time sheet with $\_ GET ["sheet\_id"]. Also deletes all time slots associated with that time sheet.

**Return**:

Result: TRUE

Error: Timesheet not found

Output: NULL

***SetActiveTimeSheet***

**Requirement(s)**: $\_GET["sheet\_id"]  
**Action**: Changes time sheet with $\_GET["sheet\_id"] to ACTIVE state and sets all other time sheets to INACTIVE state.

**Return**:

Result: TRUE

Error: Timesheet not found

Output: NULL

***ChangeSheetName***

**Requirement(s)**: $\_POST["sheet\_id"], $\_POST["name"],  
**Action**: Changes name of time sheet with $\_POST["sheet\_id"]

**Return**:

Result: TRUE

Error: Timesheet not found

Output: NULL

**Image Action** – handles all actions related to profile images for users via the User object. **All Methods require $\_SESSION['iUserId'] and $\_GET["action"]**

***CreateImage***

**Requirement(s)**: Nothing. File must be .png

**Action**: Uploads profile image for user.

**Return**

Result: TRUE/False

Error: Upload Failure

Output: NULL

***DeleteImage***

**Requirement(s)**: Nothing

**Action**: Deletes profile image for user.

**Return**:

Result: TRUE

Error: NULL

Output: NULL

***ViewImage***

**Requirement(s)**: Nothing

**Action**: Displays profile image for user.

**Return**: a png file or

Result: FALSE

Error: No uploaded png

Output: NULL

**Chat Action** – handles all actions related to the chat room for a meal via the MealChat object. **All Methods require $\_SESSION['iUserId'] and $\_GET["action"]**

***CreateMessage***

**Requirement(s)**: $\_POST["token"] OR $\_POST["user\_name"] AND $\_POST["message"]

**Action**: Creates message with for a meal or a specific relationship between users. The relationship must be in the ACCEPTED state.

**Return**:

Result: TRUE/FALSE

Error: Meal not found, Account not found, Relation not found

Output: Message ID

***ListMessages***

**Requirement(s)**: $\_POST["token"] OR $\_POST["user\_name”].Optional: $\_GET["mes\_id"]

**Action**: List messages in a chatroom for a meal or relationship. The relationship must be in the ACCEPTED state. Displays information about each message (user name, first name, last name, message, mes id)

**Return**

Result: TRUE

Error: Meal not found

Output: (user name, first name, last name, message, mes id)