**Login and Registration:**

Registration is a two-step process. Users initially ‘apply’ for membership.

Login:

With ID and password:

1. User is presented with traditional login and password screen
2. User supplies user id and password
3. Credentials are verified and user logs in

Note: The login page also has links to recover forgotten passwords and to log in using Facebook or Google

Sign Up: (PC and mobile views)

Register user and company, capturing all appropriate profile info. This

Information can be captured from Google or Facebook using an oath-based

Register using Facebook/Google info. Subsequent screens capture shop profile.

Prospective users will be asked to identify company affiliation by selecting

from a list of already-registered companies, or entering a new company profile.

1. User is at Login Screen and clicks on the register link. If the user clicks on register via facebook or google, they will go through the Oauth login for the appropriate service. What info that is available will be captured, and they will be ushered to the profile entry screens with that info pre-entered in the appropriate fields.
2. User goes to profile entry screen. This screen has entry’s for all the usual profile info as well as a buyer/seller/customer choice, a company profile link, and password entry fields
3. If the user chooses customer, the a pro-forma company record is entered and permissions are set that prevent access to the wholesale stuff
4. If the user chooses buyer or seller, the next screen they see will be a identify company screen in which the user can either select a company with which they are affiliated or go to a new company profile screen. If the user selects a company, a message is sent to the store-admin asking for approval before the user has order permissions. This message also has a pulldown specifying the user’s company role.
5. The company profile screen includes billing info and shipping info links if the user is a seller. These must be entered if the user is given order processing permissions.

Back-end API’s involved:

$User->addUser($info)

$User->addUserToCompany($user, $company,$companyRole)

$User->oathFacebookRegInfo($facebookId)

$User->oathGoogRegInfo($googleId)

$ShipInfo->addShipInfo($companyId, $newInfo)

$BillInfo->addBillInfo($companyId, $newInfo)

Note: After the initial registration process, the user has ‘guest’ access. This entails read only access to product catalogs, the message subsystem, user profile screens. A guest user can place an order going through the normal payment processor, but cannot access sell anything. In the process of buying something, guest users will have to enter shipping information, or designate their entered name and address information as the shipping address.

What an un-approved user can’t do:

1. Place anything for sale – i.e. access the Orders tab in the PC interface.
2. Add new products or create catalogs.

Forgot username/password: (PC and mobile views)

Send email to user containing one-time key that allows access

To change password screens

1. User is at the login screen
2. User clicks on “forgot password” screen
3. User is shown a screen asking for email address and enters it
4. Email address is checked to see if it is in database. If not, User is shown a ‘not on file screen”
5. Email is sent to address provided containing one-time key
6. User clicks on the one-time key or enters it into browser
7. User is shown a new password screen
8. New password is checked to see if it is secure
9. Password is saved
10. User is shown a screen prompting login with new password

$User->getUserProfile($userName)

$User->editUserProfile($user, $editInfo)

$User->checkPasswordSecure($password)

$Message->sendEmail($userEmail)

$User->updatePassword($userId, $newPassword)

**Profile:**

The profile section deals with capturing and editing user information. The user is le=d through it at registration time, and it is available to buyers and sellers as a tab in the PC interface, or as a swiped screen in the mobile interface.

Create/edit/store profile: (PC and mobile views)

User enters profile information for store. Creator of store profile becomes

store-admin until that is changed. Store-admin ok’s registration requests

from prospective users identifying affiliation with that store, and sets

company role for these people.

(see description of initial profile entry above Sign up for initial profile entry)

1. In PC interface, user clicks on “My Profile” tab at top of screen
2. In phone interface, User clicks on profile icon at the bottom.
3. Profile Screen is displayed with the available info filled in the respective fields.
4. User touches or clicks on save button to update information

Back-end API’s involved:

$User->getUserProfile($userName)

$User->addUser($info)

$User->editUserProfile($user, $editInfo)

Add Company Profile:

1. User is at profile screen.
2. User clicks on ‘Company Profile’ Link
3. User is presented with a list of companies already registered in system with an option to skip if they are part of a company that hasn’t do business with Joonley up to now.
4. If the user selects an existing company, an approval message is sent to the store-admin who has to approve it before the company registration is complete.
5. If the user selects ‘skip’, they are sent to a company profile screen.
6. User enters company data.
7. Links to Billing and Shipping info are available.
8. User clicks ‘next’ button
9. User is presented with a list of terms each of which has a checkbox. This is used to select the ‘default terms’ that will be presented with every order entry screen.
10. User click save button, and the information is saved and sent to the admin for approval.

Preview Store Profile:

Shows all the info the system has on that store.

Company->getCompanyByName($companyName) - change: return obj not id

User is an admin:

1. User clicks on the admin tab and has the tabs expanded to display the shops tab. User clicks on that.
2. User is shown a scrolling list of all stores with the usual find box.
3. User finds the appropriate company and clicks on it
4. User is shown store profile in read-only mode
5. At the bottom of the screen are buttons for Users, Orders, Catalogs ??
6. If the user clicks on users, is shown a scrollable list of users associated with that shop.
7. If the use drills into users, is ahown a profile page for that user, At the bottom is a button that will drill into the user’s order activity.
8. If the user clicks on Orders is shown a scrollable list of order activity.

Add/Edit payment Info:

Sets a company’s payment pref’s and identifies payment processor.

1. User is at profile entry screen or company profile screen
2. User clicks on Billing Information
3. User is directed to Billing Info screen. User can check – ‘same address as my profile’ box. Info is filled into the fields.
4. User can enter different address if there is a different address.
5. User clicks on ‘next’ – Entered info is saved, and user returns to main profile screen or company profile screen (depending on where he/she came from.

Back-end API’s involved:

BillInfo->addBillInfo($companyId, $newInfo)

BillInfo->editBillInfo($billInfoId, $editInfo)

BillInfo->removeBillInfo($billInfoRecordId)

BillInfo->getBillInfoById($billInfoId)

BillInfo->getBillInfoForCompany($companyId)

(BillInfo may need chaging when cc processing is determined)

Add/edit shipping info:

1. User is at profile entry screen or company profile screen
2. User clicks on Shipping Information
3. User is directed to Billing Info screen. User can check – ‘same address as my profile’ box. Info is filled into the fields.
4. User can enter different address if there is a different address.
5. User clicks on ‘next’ – Entered info is saved, and user returns to main profile screen or company profile screen (depending on where he/she came from.

Note: in the new registration process, User is led through all of these screens in a series of ‘continue’ links. For example, after filling in the initial user information, the user is prompted ‘Enter Company Info ?’ There’s an Ok/Skip it button. If Skipit is clicked, the user is asked ‘Use my Profile Address for Shipping ? For Billing’. Following this is a screen where the user is asked to specify payment defaults. There is an opt-out – ‘I will supply then when I buy something’. If the user wishes to enable 1-click ordering, CC info will have to be entered. There should be a message describing how the information is kept encrypted in the system.

**Shopping:**

The shopping section deals with features available to buyers that support buying things. It appears as a tab in the PC interface, and swiped screen in the mobile app.

Specials:

Specials are handled in two ways by the system – products are that part of a special are contained in a collection that is typed as a specials collection. Specials prices, terms, and expiration dates are to be embedded in the ContainedAs record. Specials will be stored as messages that are also appropriately typed.

Specials may also be highlighted in the Joonley Magazine/Blog, but in that case, they would need to be really special.

View Current Specials:

View products that are contained within collections of type specials.

1. User clicks or touches the shopping tab
2. In response to the ‘Shop Where ?’ propmpt, the customer chooses specials
3. Scrollable list of products contained in special collections is displayed
4. User can click on an individual product
5. User can click on ‘order this’ button – transition to order process.

API’s involved:

$Product->getProductsInCollection($collectionId, $categoryFilter)

$product->getProductsInSpecialCollections($categoryFilter)

Get Participating Stores:

View a list of stores with products in specials collections.

1. User is looking at specials list
2. User clicks on ‘see specials shops’ button
3. User is shown a list of shops having items contained in specials collections.

API’s involved:

$Collections->getCompaniesHavingCollection($collectionId)

View news, highlighted specials, and updates:

One of the ways joonley can highlight curated products is to present a website/blog featuring media-heavy glossy content on identified products that serve as a kind of ‘front door’ to the commercial site. These articles would contain ‘buy this’ links that convey the user directly to parts of the system that enable ordering. This feature can be supported by any one of a number of available blogging packages.

1. User doesn’t even need to be logged in, but can log in via ‘Log In to Joonley Store’ link
2. User sees something they like
3. User clicks on ‘buy me’ link
4. User is conveyed to login screens if he/she is not yet logged in, and then to an order page for the product.

Upload web pages for promotional pages:

1. User is admin
2. User identifies the folders containing the articles
3. User uploads the content.
4. User adjusts the entry routes to support access to the newly added pages.

Shop Catalogs:

Note: A shop can have several catalogs, among them a wholesale catalog (shown only to buyers) and a retail catalog (shown to buyers and retail customers).

Find products to purchase.

1. User clicks or touches the shopping tab
   1. The user chooses ‘Shop Where ?’. User can then find a shop by variety or means.
      * 1. The user can click on shop location and will be asked for zipcode
        2. The user can ‘By product catgories’ and is usehered through the category tree
        3. The user can select shop name and will be asked to enter a shop name.
        4. If the User selects shop from favorites, he/she is presented with the list of favorite shops. User selects a shop and is then shown the list of products from the shop current published catalogs. There will be a filter pulldown to allow the user to select particular catalogs.
        5. If the user selects ‘specials’ is shown products on special (see above)
        6. If the user selects ‘marketplaces’ he/she will be shown a list of marketplaces.
      1. When a shopping place (i.e. a shop) has been found the shop’s current published catalog will be displayed
      2. There will be an ‘add to favorites’ button on any of the shop pages.
   2. The user chooses ‘Shop for What ?’
      1. User selects ‘explores categories’ and goes down the category tree. At every juncture, user can click on ‘see products’ button. A number is provided within the button indicating how many products are in that category.
         1. At the leaf level, user is ushered to a scrolled list of products. Mini icons appear next to the product name. Upon mounse-over, the mini-icons blow up to bigger pictures
      2. User selects ‘Search by keyword’
         1. Entry field appears
         2. User types in keyword(s)
         3. Matches on product descriptions is run and hits are displayed
      3. User clicks on the picture and is ushered into the product details screen.

API’s involved:

$NestedCategory->findChildNodes($parentNodeName)

$Collection->getCollectionByCompanyAndType($company, $collectionType)

$Company->findCompaniesByLocation($location)

$Company->findCompaniesByProductTypes($productTypes)

Ask for a sample:

1. User must be typed as a company-associated buyer to have this option available.
2. User must have arrived at a product screen
3. User clicks on ‘ask for a sample’
4. Message box appears
5. User types in a message to the product’s seller.
6. Message is sent with information on the product requested automatically included.

API’s involved:

$Message->addNewMessage($msginfo)

Buy product in View:

1. User (Buyer) is at a product screen or in the Joonley magazine.
2. If the user is in the magazine and hasn’t logged in, he/she is routed through the login screens.
3. User clicks on buy this button
4. Order screen is displayed with all the order info filled in
   1. Pulldowns will appear for each set of the options that have been selected from the default options when the product was originally entered.
   2. If user fails to select an option prior to clicking the ‘add’ button, then a dialog screen will appear to get that information.
5. User is asked for shipping type
6. User clicks on order button
7. If user doesn’t have shipinfo filled in a dialog appears to capture the ship info.
8. Terms for this product are displayed
9. User is ushered to the payment processing service
10. User is show ‘Message Seller ?’ dialog.
11. Order is placed with message to seller created.

API’s involved:

$Orders->startNewOrder($company, $user)

$Orders->addProductToOrder($orderId, $productToAdd, $optionsSelected, $thisShipTypeId, $quantity)

$Orders->getIncompleteOrders($companyId)

$Orders->placeOrder($orderId)

$Orders->selectPayment($orderId) - Need to finalize payment processing

$Orders->selectShippingOption($orderId)

Start an order:

1. User clicks or touches the shopping tab
2. User touches/clicks on ‘Start an Order’
3. User is shown a ‘To What Shop ?’ dialog
   1. Locate by name option selected – use fills in name of shop
      1. If shop is found, user proceeds to the order page with seller info shown at top
      2. If shop is not found, user is shown a list of similar shop names
      3. If shop is still not identified, seller can go back and pick another way of identifying the seller
   2. Pick from favorites
      1. User’s shop favorites is shown
      2. User selects a shop from the list
   3. Pick by Product category
      1. User is directed through the prduct category tree
      2. At any level, user can click on ‘see shops that sell this’
      3. User is shown a list of shops
      4. User picks a shop from the list
4. User is shown an empty order form
5. User clicks on ‘add something to this order’
6. User is shown the shop’s product catalog
7. User selects a product
8. User selects the appropriate options from the options pulldown
9. User selects a quantity
10. User selects shipping option for this item.
11. User clicks on ‘add this to the order’ button
12. User is show the order with the product just added as one of the line items. The current order total is shown at the bottom along with the terms.

API’s involved:

$Orders->startNewOrder($company, $user)

$Orders->addProductToOrder($orderId, $productToAdd, $optionsSelected, $thisShipTypeId, $quantity)

Finalize an order:

1. User is on an in-progress order page.
2. User clicks on order button
3. If user doesn’t have shipinfo filled in a dialog appears to capture the ship info.
4. Terms for this product are displayed
5. User is ushered to the payment processing service
6. User is show ‘Message Seller ?’ dialog.
7. Order is placed with message to seller created.

API’s involved:

$Message->addNewMessage($msginfo)

$Orders->placeOrder($orderId)

$Orders->selectPayment($orderId) - Need to finalize payment processing

$Orders->selectShippingOption($orderId)

Cancel Order:

Order status set to canceled. Order not deleted and can be revived later

1. User is at an order entry screen
2. User click on the ‘cancel order’ button
3. User is shown an order summary screen with an ‘are you sure’ dialog.
4. User clicks on ‘ok’ button
5. Order’s status is set to ‘canceled’ and no longer appears on the user’s ‘My Orders’ list.

$Orders->setStatusCanceled($orderId)

Return To Saved Order:

List to canceled orders is displayed. Order can be set to status in-progress

1. User is at ‘My Orders’ screen
2. User clicks on ‘show canceled orders’
3. User selects order he/she is interested in
4. User is shown order summary screen
5. User clicks on ‘re-activate this order’ button
6. Order’s status is changed to ‘in-progress’
7. Order now appears on the regular ‘my orders’ screen.

API’s involved:

$Orders->getLineItems($orderId)

$Orders->getOrderAll($orderId)

Orders-getCanceledOrders($companyId

Place digital order at show:

Capture barcode from camera. Include in favorites list or order.

1. User is at a show looking over products at a Joonley seller’s both. The user notices a product he/she is interested in. There is a label on the product with a barcode.
2. User clicks on photo icon, pointing camera at product’s barcode
3. If user is not at order screen with an in-progress order, a dialog appears asking ‘add product to order’ or ‘add product to favorites’.
4. If the response is to add product to favorites, the product is added to favorites and the user goes back to whatever he/she was doing previously.
5. If the user is at an order screen, the camera captures the barcode, looks up the product, and adds it as a line item to the order.
6. User completes the options and quantity info
7. User clicks on ‘add item to order’ button.
8. The order is saved as an open order.

API’s involved:

$Company->findCompanyFromCameraBarcode()

$Product->findProductFromCameraBarcode()

$Orders->addProductFromCameraBarcode()

$Orders->placeOrder($orderId)

Add Shop to Favorites:

1. Two ways of accessing this feature:
   1. User clicks/touches Shopping tab
   2. User selects ‘Shope Where ?’ option
   3. See 1) a) above.
2. User is filling in an order.
   1. User clicks ‘add this shop to favorites on order page
3. User is looking at specials
   1. User has navigated to a special
   2. User can click on add this store to favorites

API’s involved:

$Company->saveCompanyToFavorites($companyId)

Manage Orders:

View existing orders with specified status.

1. User clicks/touches Shopping tab
2. User selects ‘My Orders’ option
3. User can:
   1. View Open Orders
   2. View Confirmed Orders:
   3. View Shipped Orders:
   4. View Declined Orders:
   5. View Canceled Orders
4. User is ushered to a list of orders which have a status corresponding to what they selected. In the case of open orders, at the bottom of the list appears a short summary detailing the total cost of existing open orders.
5. User can select an individual order from the list and is ushered to an order page. Depending on the status of the orders, buttons will be available that permit the user to:
   1. Open Orders
      1. User can add to edit, finalize, or cancel the order
         1. User clicks on edit button
         2. Order-level properties – shipping info are displayed in editable fields
         3. User can click on any one of the line items, bringing up an editable screen where any of the details of the line item can be changed
         4. User can click on the ‘remove this item’ to remove the item from the oder. There will be an ‘are you sure dialog’
         5. Back at rthe order screen, there will be a ‘save changes’ button that will cause the edited order to be saved.
         6. There will also be an ‘add line item’ button that allows the user to add items to the order.
   2. Confirmed Orders
      1. User can message the seller or cancel the order if it has not been shipped.
      2. Reorder:
         1. User is at the order screen of a confirmed order
         2. User clicks on the ‘re-order’ button
         3. The order is duplicated in its entirety and displayed as an open editable order (see edit order above)
         4. User can edit order as if it were any open order
         5. User can place the order by clicking on the ‘place order’ button.
   3. Declined Orders
      1. User can message the seller, edit, resubmit or cancel the order
      2. Yser can click on ‘clear this order’ button that results in it being removed from the system.
   4. Canceled Orders:
      1. User can re-active the order and turn it into an open, in-progress order.

API’s involved:

$Orders->getOrdersByStatus($companyId, $status)

$Orders->getCompanyOrdersByStatus($company, $status)

$Orders->getUserOrdersByStatus($user, $status)

$Orders->getLineItems($orderId)

$orders->setStatusCanceled($orderId)

$Orders->removeStartingOrder($orderId)

$Message->addNewMessage($msginfo)

$Orders->editOrder($orderId)

$Orders->removeLineItem($OrderId, $lineItemId)

$Orders->duplicateOrder($orderId)

**Messages:**

Initiate nessage with shop:

Send message to shop-admin

1. User is at Product screen or Shop screen
2. User clicks on ‘message the seller’ button
3. Message box appears and user enters the message
4. User clicks on send button. A new thread is created.

$Message->addNewMessage($msginfo)

Access message center:

See all your current messages.

1. User clicks on mail icon at the bottom
2. A scrollable list of all unread thread headers is displayed
3. User can click on an individual message to read it.

$Message->getMessagesToUser($user) – need to add filter

See previous message:

Page to previous message.

1. User is at an individual message screen
2. User clicks on up arrow icon
3. Previous thread is displayed

$Message->getPreviousMessage($messageId)

See next message:

Page to next message.

1. User is at an individual message screen
2. User clicks on down arrow icon
3. Next thread is displayed

$Message->getNextMessage($messageId)

Get All Message Headers in thread

See all message headers in current thread.

1. User is at the message screen
2. Thread’s initial message is displayed at top
3. Headers are below the message

$Message->getMessagesInThread($messageId)

Add message to thread:

Add a reply message to current thread.

1. User is at the message screen
2. User clicks on ‘plus’ icon
3. Message box appears
4. User enters text
5. User clicks on save button
6. Message is added to the thread.

$Message->addMessageToThread($messageId, $msginfo)

Add picture to message:

Upload or add camera picture to current message

1. User is at a message entry box
2. User clicks on upload icon – file browser appears, file uploaded and attached to the current message
3. Use clicks on the camera icon
4. Camera takes picture
5. Picture is uploaded and attached to the message.

$Message->uploadPicture()

$Message->addPicture($messageId)

Mark thread:

Mark a thread read or unread. Will revert to read if new reply added.

1. User is at the individual message screen
2. User clicks on ‘mark this thread as read
3. Thread will no longer appear in list of messages unless user clicks on see all messages button in the message center.

$Message->markThread(readUnreadStatus)

Delete Message:

Remove a message from the system entirely.

1. User is at individual message screen
2. User clicks on delete this thread button
3. Thread is removed from the system perminantly

$Message->removeMessage($messageId)

**Catalogs/Products**

These features will be made available only to users that are sellers representing a store.

Note: Every store has an ‘all products’ catalog which contains all the products a store has. This catalog cannot be deleted, or published.

Show store catalogs:

1. User is a store-admin
2. User clicks/touches Shopping Tab
3. User select ‘My Store’
4. User selects ‘My Catalogs’
5. List of Catalogs is displayed with each entry indicating its type and whether it is published or not.
6. User can click on an individual Catalog to drill down to a list of products
7. User can select an individual product in a catalog and preview the display as well as get sales figures etc.

API’s involved:

$Collections-> getCollectionsByCompanyId($companyId)

Create a new Catalog:

1. User is a store-admin
2. User clicks/touches Shopping Tab
3. User select ‘My Store’
4. User selects add catalog
5. Entry page for new catalog is shown
6. User clicks on ‘save’ button. Catalog will now appear in the list of catalogs

API’s involved:

$Collections-> addCollection($collectionName, $collectionSlug,$collectionDescription,$collectionType,$collectionStatus, $collectionCompany)

Delete a Catalog:

1. User is a store-admin
2. User clicks/touches Shopping Tab
3. User select ‘My Store’
4. User selects ‘My Catalogs’
5. User selects an individual catalog from the list other than the ‘all products’ catalog
6. User can click on ‘delete catalog’ button. The catalog will be deleted.

API’s involved:

$Collections->deleteCollection(collectionId)

Add Product:

1. User is a store-admin
2. User clicks/touches Shopping Tab
3. User select ‘My Store’
4. User selects ‘Add a Product’
5. User fills in product information.
6. User identifies product type
7. User selects options available for the product
8. User selects terms
9. User can upload picture of product
10. User can use phone camera to take a picture of the product
11. User can select catalogs to add product to.
12. User clicks on ‘save product’ button and product is added by default to ‘all products’ catalog

API’s involved:

$Collections-> addProductToCollection($productId, $collectionId, $containedAsId)

Product->addProductUsingDefaults($productInfo)

Options->linkDefaultOptionsToProduct($productType, $productId)

Terms-> addDefaultTermsToProduct($companyId, $productId)

Remove a product from a catalog:

1. User is a store-admin
2. User clicks/touches Shopping Tab
3. User select ‘My Store’
4. User selects ‘My Catalogs’
5. User selects an individual catalog from the list
   1. If user selects ‘all catalogs from list, removing a product removes it from all of the store’s catalogs – removes it altogether.
   2. If a user selects any other catalog, the removal simply severs the link of that product to that particular catalog and it remains in the ‘all products’ catalog
6. User selects a product from list of products displayed as being associated with that catalog
7. User clicks on the ‘remove this product’ button
8. An ‘are you sure’ dialog appears (see 5a and 5b above.
9. Product is removed.

API’s involved:

$Collections->removeProductFromCollection(($productId, $collectionId)

Edit a product:

1. User is a store-admin
2. User clicks/touches Shopping Tab
3. User select ‘My Store’
4. User selects ‘My Catalogs’
5. User selects ‘All products’
6. User selects product to be edited
7. User makes changes – uploads new photo, selects different options or terms etc
8. User clicks on ‘Save Product Information’ button

$Collections-> addProductToCollection($productId, $collectionId, $containedAsId)

$Collections->removeProductFromCollection(($productId, $collectionId)

Options->linkDefaultOptionsToProduct($productType, $productId)

Terms-> addDefaultTermsToProduct($companyId, $productId)

Options->setOptionsForProduct($options, $product)

Terms->setTermsForProduct($terms, $product)

**Admin**

Note: This set of features will only be available on the PC interface.

Add a product category.

1. User is admin
2. User clicks on the admin tab.
3. User selects add category from side menu-bar
4. Top level of the category is displayed
5. User clicks down to through the category tree until the point where the new category to be added will be a child of an existing category
6. User clicks on ‘add category’ button
7. User fills in new category label and description information
8. User clicks on add category button
   1. If there are existing products in the parent category, User is prompted as to whether stores with products in this category should be messaged.
   2. User fills in message notifying stores that a new category has been added and they should consider changing their product’s catagorizartion information
9. Category is added

API’s involved:

$NestedCategory-> findChildNodes($parentNodeName)

$NestedCategory->addCategory ($name, $description, $parentNodeName)

$Message-> addNewMessage($msginfo)

Remove a category

1. User is admin
2. User clicks on the admin tab.
3. User selects add category from side menu-bar
4. Top level of the category is displayed
5. User clicks down to through the category tree until the category to be deleted is reached
6. User clicks on ‘remove this category’button
   1. If there are products in this category, the user is prompted to identify a new category that these products should be moved to.
   2. If there is category change for any products, the store owners will be automatically messaged.

API’s involved:

$NestedCategory-> findChildNodes($parentNodeName)

$NestedCategory->deleteCategory($nodeName)

$Message-> addNewMessage($msginfo)

Disable a User Account:

1. User is an admin
2. User clicks on admin tab
3. User select ‘disable a user login’ from side menu
4. User enters user id
5. User clicks ‘disable this account’ button
6. An ‘are you sure dialog’ appears
7. User clicks on ‘disable’ button
8. Account login is disabled

API’s involved:

$User->disableLogin($userId)

Re-Enable an Account:

1. User is an admin
2. User clicks on admin tab
3. User select ‘disable a user login’ from side menu
4. User enters user id
5. User clicks ‘re-enable this account’ button
6. An ‘are you sure dialog’ appears
7. User clicks on ‘re-enable’ button
8. Account login is re-enabled

API’s involved:

$User->enableLogin($userId)

Remove a store:

1. User is an admin
2. User clicks on admin tab
3. User selects ‘remove a store’ from menu side-bar
4. User is shown a list of stores
5. User selects store to be removed
6. User clicks on ‘remove this store’ button
7. An ‘are you sure’ dialog appears
8. User clicks on ‘remove’ button
9. The store is removed along with all its products, collections and order records. These are archived and can be re-instated at a later date.

API’s involved:

$Company->removeCompany($companyId)

$Company->archiveCompanyInformation($companyId)

$Company->restoreCompanyInformation($fileLocation)

Disable a store:

1. User is an admin
2. User clicks on admin tab
3. User selects ‘disable a store’ from menu side-bar
4. User is shown a list of stores
5. User selects store to be disabled
6. User clicks on ‘disable this store’ button
7. An ‘are you sure’ dialog appears
8. User clicks on ‘disable store’ button
9. All users connected with that company have their logins disabled.

API’s involved:

$Company->disableCompany($companyId)

$User->disableLogin($userId)

Re-enable a store:

1. User is an admin
2. User clicks on admin tab
3. User selects ‘re-enable a store’ from menu side-bar
4. User is shown a list of stores that have been disabled
5. User selects store to be disabled
6. User clicks on ‘re-enable this store’ button
7. An ‘are you sure’ dialog appears
8. User clicks on ‘re-enable store’ button
9. All users connected with that company have their logins re-enabled.

Get approval/rejection: (PC and mobile views)

Registration requests come to admin as a message of a particular type.

User must be an admin.

1. Admin is in message center screen and clicks on message identified as a registration request.
2. The message contains a summary of the profile info and a ‘show details’ button as well as an approve/decline button. The message also has a ‘remove user’ button which terminates the process.
3. If the admin clicks on the show details button, the admin is shown the profile screen in view-only mode
4. The message also has an entry area for a reply
5. If the admin approves, an email and a message are both sent to user. The email has a confirm key the user has to click on the confirm registration. When confirm link is activated, te user’s permissions are changed.
6. If the admin declines, email goes to user with a link to access further messageing. The email can contain info the admin types in the the message entry field contained in the request message.

$User->changeUserRole($user, $role)

$Message->getMessagesByType($userId, $filter)

$Message->sentEmailToUser($emailAddress)

$Message-addNewMessage($msginfo)

$User->removeUser($userId)

Ask questions about registration: (PC and mobile views)

Questions come as messages of a particular type.

1. Messages from applying users appear in the admin’s message center as a part of a thread.
2. Admin has option to also send emails
3. Admin’s reply screen has click boxes for frequently asked questions and requests for further information.

$Message->addNewMessage($msginfo)

$Message->sentEmailToUser($emailAddress)