Requirements

**Paywall**

Version 1

Bob Muller

December 2, 2014

643 Bair Island Road Suite 403  
Redwood City, CA 94063

# Introduction

The business of Phoenix Bioinformatics is to provide scientific data in a sustainable manner. To do so, we have determined that we need to charge scientists for access to the products we offer. To better support this approach, we want to develop a basic paywall system.

A *paywall* is a system that prevents Internet users from accessing webpage content without a paid subscription (<http://en.wikipedia.org/wiki/Paywall>). A *hard paywall* is a paywall that prevents any access without subscription; a *soft paywall* is more flexible, permitting certain kinds of unrestricted or partially restricted access without a paid subscription. Phoenix will support seven kinds of access:

1. Free content: web pages accessible by anyone with no charge
2. Login-required free content: web pages requiring authentication but not subscription
3. Paid content: web pages requiring a subscription
4. Login-Required paid content: web pages requiring authentication and a subscription
5. Metered content: paid content allowing a certain number of accesses per month before requiring a subscription (determined by IP address)
6. Curator content: web pages accessible only to authenticated curators
7. Administrator content: web pages accessible only to authenticated administrators

Each of these types of access has a use case in this document as the entry point into the system.

*Note: The first release of the paywall does not support curator and administrator authentication.*

A full-scale, commercial-quality paywall offers lots of features, including subscription management, access control, system monitoring, and system and financial reporting. It is also very expensive, usually based on a percentage of subscription revenue between 10 and 20%.

Because of some of the decisions we've made about our subscription model, we do not really need a comprehensive paywall solution. This document defines the specific use cases we need to support to achieve the level of access control to our content we want.

This document also includes the use cases for subscription processing, but not for registration and profile management or for invoice processing or other financial processes, which are outside the scope of the paywall. Also note that the assigning of Curator and Administrator roles to individual users is done outside the scope of this system in the first release.

This document also includes use cases for tracking visits, the basic measure used to determine metering status and subscription tier membership for institutions. There are two kinds of visit of interest to Phoenix that the paywall needs to track: web visits, as measured by Google Analytics, and application visits, as measured by paywall URL tracking.

Google Analytics defines a *visit* as "a group of interactions that take place on your website within a given time frame." (https://support.google.com/analytics/answer/2731565?hl=en) The TAIR JSPs implement Google visits by specifying a cookie for the various entities that comprise a visit. At the moment, a visit includes any page in TAIR, so the time period determines the visit. We will need to make sure only paid content is included in visit statistics. The data associated with a visit comprises the IP address of the visitor, the date, and a count of visits on that date. Google Analytics defines a *session* as a group of interactions that take place on your website within a given time frame. Google Analytics is currently implemented through JavaScript code in the target server, not in the paywall.

A *TAIR application visit* is the proxying of a URL to the TAIR server. The data associated with an application visit comprises a session id, the IP address, an optional community id for logged-in users, the proxy status (Proxied, Refused, Metered, Error), and the URL and query parameter string defining the servlet execution content. A *TAIR application session* is a collection of TAIR application visits with the same session id. The session id is likely to be the JSESSIONID of the Paywall application server session. The server configuration controls the length of inactive sessions.

*Note:* This document is a complete revision of the original requirements document dated March 4, 2014, revising the December 2013 document. This revision removes the Login and Registration use cases to another requirements document, as those are now TAIR functions, not Paywall functions. The requirements for Paywall Version 2 are currently underway and will expand the ability of the Paywall to support multiple backend target systems; the version documented here supports only www.arabidopsis.org.

# Allow Free Access

Certain TAIR content is available to any Internet user without subscription or login.

1. User submits a www.arabidopsis.org URI that the paywall designates as free content.
2. System responds by showing the content.

# Allow Paid Access

Certain TAIR content is available to any registered user who has paid for a subscription, either through individual or group subscription or through institutional or company subscription.

1. User submits a www.arabidopsis.org URI that does not match any URI the paywall designates as free content.
2. System uses the Check Subscription use case to validate the subscription and shows the content.

Extensions:

* If the Check Subscription use case fails, the System treats the user as an unpaid user and uses the Subscribe use case to allow the user to subscribe.

# Allow Metered Access

For paid content, the paywall allows a User a certain number of accesses per month (a technique called *metering*). This permits the User to see the restricted pages without paying a subscription.

1. User submits a www.arabidopsis.org URI that does not match any URI the paywall designates as free content.
2. System uses the Check Subscription use case to validate the subscription and fails. System gets the IP address of the User and retrieves the number of metered accesses for the user during the current month (see the Track TAIR Application Visits use case). If the metered accesses is less than or equal to the limit, the System shows the content.

Extensions:

* If the number of metered accesses greater than 7, System displays an alert informing the user that the user must subscribe after 10 accesses.
* If the number of metered accesses is greater than 10, System uses the Subscribe use case to let the User subscribe.
* At midnight on the first day of every month, System clears the metering database.

# Allow Login-Required Access

For some content, TAIR allows only an authenticated community member to access the page.

1. User requests a page. System determines that the page requires login in the Community Role. System determines whether the user is authenticated.
2. System displays the content if the user is authenticated.

Extensions:

* If User is not authenticated, System denies access to the page.
* If the content is paid or curator or administrator content, System uses those use cases as well to determine whether to display the content.

# Allow Access without Login

For some free or paid content, TAIR allows access to the page without authentication. Most pages in TAIR are login-free pages.

1. User requests a page. System determines that the page does not require login.
2. System displays the content.

Extensions:

* If the content is paid or curator or administrator content, System uses those use cases to determine whether to display the content.

# Allow Curator Access

For some login-required content, TAIR allows only an authenticated Curator to access the page. An authenticated curator can also access any free, metered, or paid content. *Implemented in the target server in the first release.*

1. User is granted the Curator role.
2. User requests a page. System determines that the page requires login in the Curator role.
3. System displays the content.

Extensions:

* If User does not have the Curator role, System displays an error page telling the user that the content is not accessible.

# Allow Administrator Access

For some login-required content, TAIR allows only an authenticated Administrator to access the page. An authenticated Administrator can access any content. *Not implemented in the first release.*

1. User is granted the Administrator role.
2. System displays the content.

Extensions:

* If User is not an authenticated Administrator, System displays an error page telling the user that the paid, curator-accessible, or administrator-accessible content is not accessible.

# Check Subscription

Under various circumstances, the System needs to get the individual user's subscription status. This is a System use case that other use cases use by reference.

1. System retrieves the authenticated user's subscription status for TAIR. If the expiration date is greater than the current date and there is a valid payment, the System validates the subscription.

Extensions:

* If the User is not authenticated, the System invokes the Login or Subscribe use case.
* If the User is authenticated but the expiration date is less than or equal to the current date, the System invalidates the subscription.
* If the User is authenticated and there is no expiration date, the System invalidates the subscription (the user is registered and can access free content that requires login, but the User did not subscribe for access to paid content).

# Subscribe

The new User subscribes to TAIR for a year or two years.

1. System displays the Subscribe to TAIR page with required fields for username, password, re-enter password, email, first name, last name, and institution name. There is a checkbox for automatic renewal, checked by default. The page specifies the subscription fee. A choice field lets the User select month (with required autorenew), one year, or two years, with a default of 2 years. System updates the fee based on the choice ($9.50/month, $95/year, $190/2 years).
2. The User fills out and submits the form.
3. System uses the Accept Payment use case to get the User to pay for the subscription. System waits for the Confirm Payment use case to complete. That use case handles sving the payment information.
4. System registers the user as a basic TAIR user, allocates a community id, and stores the user registration for the user, including the expiration.

Extensions:

* The User can choose to subscribe a group of users (a checkbox). On doing so, System uses the Subscribe Group use case.
* If the Confirm Payment use case fails to confirm payment, the System displays an error message and does not register the user.
* The User can choose to register without subscribing, in which case the System creates the user and registers them without an expiration date.
* If the username is already present (case insensitive) in the system, the System displays an error to that effect and suggests logging in to upgrade with the Upgrade Subscription use case instead.

# Subscribe Group

The user subscribes a group of users (for a discounted rate).

1. System presents a form with fields for the user's username, password, re-enter password, email address, first name, last name, institution name, and a table of similar fields (username, password, re-enter password, email address, first name, last name) for the multiple users in the group. A choice field lets the User select one or two years default 2 years. All fields are required.
2. User enters the group leader information (PI information) and enters one row in the table for each additional subscription desired. As User completes each row, System updates the total amount of the subscription fee with discount based on the subscription period choice and the number of rows. User submits the form.
3. System uses the Accept Payment use case to get the User to pay for the subscription. System waits for the Confirm Payment use case to complete.

System allocates community ids and stores the user registration for the users, including the expiration date (in the Person table and in the external authentication system). System registers the PI User with the information provided and links the other users to the PI User in the database. System uses the Accept Payment use case to get payment from the User.

Extension:

* If the Confirm Payment use case fails to confirm payment, the System displays an error message and does not register the group of users.
* If the username already exists in the system (case-insensitive), the System displays an error message to that effect and suggests logging in to upgrade instead.

# Accept Payment

The User pays for a subscription using a credit card. The System accomplishes this by passing through to an external payment system that handles the credit card processing.

1. User chooses to pay for a subscription. System displays the credit card entry form.
2. User enters data into the form as required and submits. System transmits the data and processes the payment, then receives confirmation.
3. System uses the Confirm Payment use case to inform the User that the payment was successful and to provide a receipt.

# Confirm Payment

The System confirms the details of a subscription payment to the User.

# Upgrade Subscription

The registered User not covered by an institutional or personal subscription upgrades to a personal subscription for one or two years.

1. User logs in and is authenticated. User does not have an existing subscription and is not covered by an institutional subscription (the current IP address is not in a subscribed IP range). User chooses to subscribe.
2. System displays the Subscription page with fields for username, email, and institution name filled in with the logged-in User information. There is a checkbox for automatic renewal, checked by default. The page specifies the subscription fee. A choice field lets the User select month (with required autorenew), one year, or two years, with a default of 2 years. System updates the fee based on the choice ($9.50/month, $95/year, $190/2 years).
3. The User makes any changes and submits the form.
4. System uses the Accept Payment use case to get the User to pay for the subscription. System waits for the Confirm Payment use case to complete.
5. System sets the expiration date to one or two years from the current date.

Extensions:

* The User can choose to renew a group of users (a checkbox). On doing so, System uses the Subscribe Group use case.
* If the Confirm Payment use case fails to confirm payment, the System displays an error message and does not upgrade the user and sets the expiration date to null.

# Upgrade Group Subscription

A registered User principal investigator (PI) upgrades a group of registered but unsubscribed users for a discounted rate. The monthly subscription term has no discount.

1. The PI User logs in and is authenticated.
2. System presents a form with filled-in fields for the user's username, email address, first name, last name, institution name, and a table of similar filled-in fields (username, email address, first name, last name) for all affiliated but unsubscribed users (that is, users with no expiration date) in the group. There is a checkbox for automatic renewal, checked by default. A choice field lets the User select one month (with required autorenew) or one or two years, default 2 years. All fields are required.
3. User can either add rows or delete rows. An added row results in a new User added with an affiliation to the PI and his lab. A deleted row results in an affiliated User not being upgraded. As User completes each row, System updates the total amount of the subscription fee with discount based on the subscription period choice and the number of rows. User submits the form.
4. System uses the Accept Payment use case to get the User to pay for the subscription. System waits for the Confirm Payment use case to complete.
5. System allocates community ids and stores the user registration for any added users, including the expiration date (in the Person table and in the external authentication system), linking those users to the PI. System updates the expiration date for the PI user and for the upgraded users to the selected subscription term from the current date.

Extension:

* If the subscription term is one month, there is no discount.
* If the Confirm Payment use case fails to confirm payment, the System displays an error message and does not register the group of users.
* If a new (added) User in the group specifies a username (case insensitive) that already exists, System displays an error message. Note this is not an upgrade situation, just a duplicate username.
* If there are no unsubscribed affiliated users for the PI User, System displays a blank record to allow the PI to add more users.
* If the PI User is already subscribed (has an expiration date in the future), then the total amount excludes the appropriate subscription term amount, and on submission the System does not update the PI User expiration date.

# Initiate Subscription Renewal

If a user has an individual subscription and has de-selected the auto-renew option, we need to send them a renewal reminder a month before their subscription expires.

1. System queries all subscribed users with autorenew set off.
2. For each user, System sends an email to the user with a standard reminder to renew their subscription by logging in and going to the Renew Subscription page (link provided).

# Renew Subscription

The subscribed User renews their subscription to TAIR for a year or two years.

1. User logs in and is authenticated.
2. System displays the Subscription page with filled-in fields for username, email, first name, last name, and institution name. There is a checkbox for automatic renewal, checked by default. The page specifies the subscription fee. A choice field lets the User select month (with required autorenew), one year, or two years, with a default of 2 years. System updates the fee based on the choice ($9.50/month, $95/year, $190/2 years).
3. The User fills out and submits the form.
4. System uses the Accept Payment use case to get the User to pay for the subscription renewal. System waits for the Confirm Payment use case to complete.
5. System updates the expiration date to add one or two years to the existing date.

Extensions:

* The User can choose to renew a group of users (a checkbox). On doing so, System uses the Subscribe Group use case.
* If the Confirm Payment use case fails to confirm payment, the System displays an error message and does not renew the user, leaving the existing expiration date as is.

# Renew Group Subscription

The PI User renews a group of users (for a discounted rate).

1. The PI User logs in and is authenticated.
2. System presents a form with filled-in fields for the user's username, email address, first name, last name, institution name, and a table of similar fields (username, email address) for the affiliated users with expiring or expired subscriptions in the group. There is a checkbox for automatic renewal, checked by default. A choice field lets the User select one month (autorenew required) or one or two years default 2 years. All fields are required.
3. User can either add rows or delete rows. An added row adds a new User and subscribes them. A deleted row removes a User from the subscription renewal list and ignores it. As User completes each row, System updates the total amount of the subscription fee with discount based on the subscription period choice and the number of rows. User submits the form.
4. System uses the Accept Payment use case to get the User to pay for the subscription. System waits for the Confirm Payment use case to complete.
5. System allocates community ids and stores the user registration for any added users, including the expiration date (in the Person table and in the external authentication system), linking those users to the PI. System updates the expiration date for the PI user and for the existing users to the existing date plus the new subscription term.

Extension:

* If a new (added) User in the group specifies a username (case insensitive) that already exists, System displays an error message. Note this is not an upgrade situation, just a duplicate username.
* If the Confirm Payment use case fails to confirm payment, the System displays an error message and does not register the group of users.

# Track TAIR Application Visits

The paywall will track TAIR application visits individually in real time. An *application visit* is a single execution of a URL in the proxy server.

1. User submits a request to the paywall with a URL and query parameters.
2. System records the visit by storing the IP address, the complete URL (including query parameters), the visit resolution status (Proxied or Refused), the optional community id for a logged-in user, and the start time and end time of the visit.

Extensions:

* If metering is in progress as determined with the Allow Metered Access use case, System increments the metered access count for the month along with the tracking information. The visit resolution status is Metered.
* If a processing error occurs, or the target server returns an HTML error return code, the visit resolution is Error.

# Paywall Processing Logic



# Paywall Schema

