Security Issues in Web Programming (Part 4)

Robert M. Dondero, Ph.D.
Princeton University

Objectives

- We will cover:
  - Data storage attacks
  - Data comm attacks
  - Third-party authentication (briefly):
    - CAS authentication
    - Microsoft EntraID authentication
    - Google authentication
    - Auth0 authentication

Agenda

- **Data storage attacks**
- Data comm attacks
- Third-party authentication (briefly)
  - CAS authentication
  - Microsoft EntraID authentication
  - Google authentication
  - Auth0 authentication

Data Storage Attacks

- **Problem:**
  - PennyAdmin app stores passwords in DB
  - If attacker gains access to DB
  - … Then attacker learns passwords

Data Storage Attacks

- **Insight:**
  - PennyAdmin doesn’t really need to store passwords
  - It’s sufficient for PennyAdmin to know if a given password is correct

How can PennyAdmin know if a given password is correct if passwords are not stored in its DB? (No fair looking at the next slide.)

Data Storage Attacks

- **Solution:**
  - Store ***password hash codes*** instead of passwords
    - `hash_code = hash(password)`

Data Storage Attacks

- Which hash function?
  - ***md5***?
    - `hash_code = md5(password)`
    - No! See https://en.wikipedia.org/wiki/MD5
  - ***sha256***?
    - `hash_code = sha256(password)`
    - Yes! See https://en.wikipedia.org/wiki/SHA-2

Data Storage Attacks

- See **PennyAdmin8aHash** app
  - runserver.py
  - **penny.sql**, penny.sqlite
  - database.py
  - header.html, footer.html
  - index.html, show.html,
  - add.html, delete.html, reportresults.html
  - login.html, signup.html, loggedout.html
  - top.py, penny.py, **auth.py**

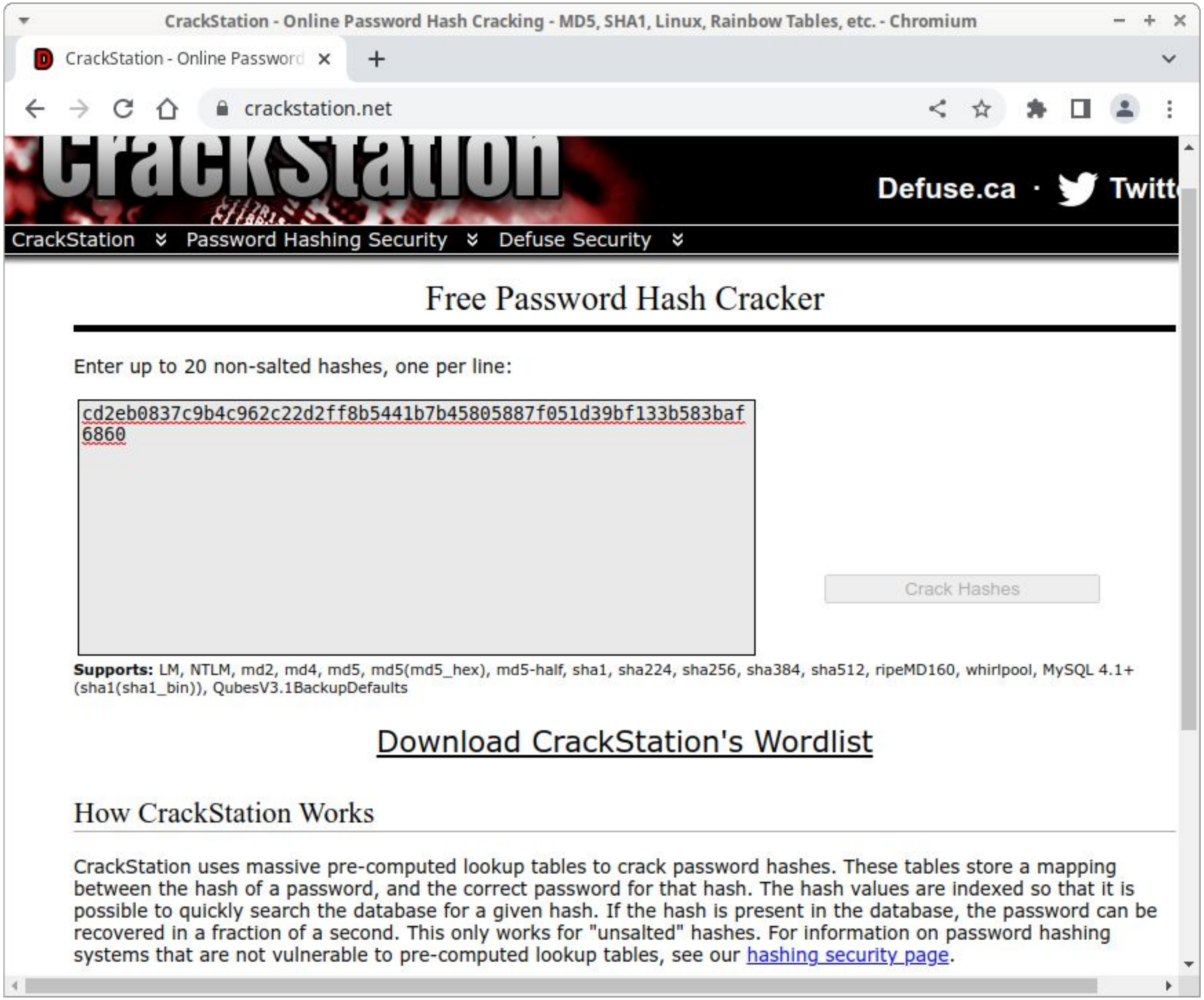
Data Storage Attacks

- **Problem:**
  - PennyAdmin app stores password hash codes in DB
  - If attacker gains access to DB, then…
    - Attacker learns password hash codes
  - If a password is common, then…
    - Attacker might find password hash code in a ***rainbow table*** (huge malevolent list of hash codes), and thereby learn the password

Data Storage Attacks

- **Example:**
  - Password:
    - xxx
  - sha256 hash code of that password:
    - cd2eb0837c9b4c962c22d2ff8b5441b7b45805887f051d39bf133b583baf6860
  - See https://crackstation.net/
    - Can derive xxx

Data Storage Attacks



CrackStation - Online Password Hash Cracking - MD5, SHA1, Linux, Rainbow Tables, etc. - Chromium
CrackStation - Online Password
crackstation.net
CrackStation
Defuse.ca · Twitt
CrackStation
Password Hashing Security
Defuse Security

Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

```
cd2eb0837c9b4c962c22d2ff8b5441b7b45805887f051d39bf133b583baf
6860
```

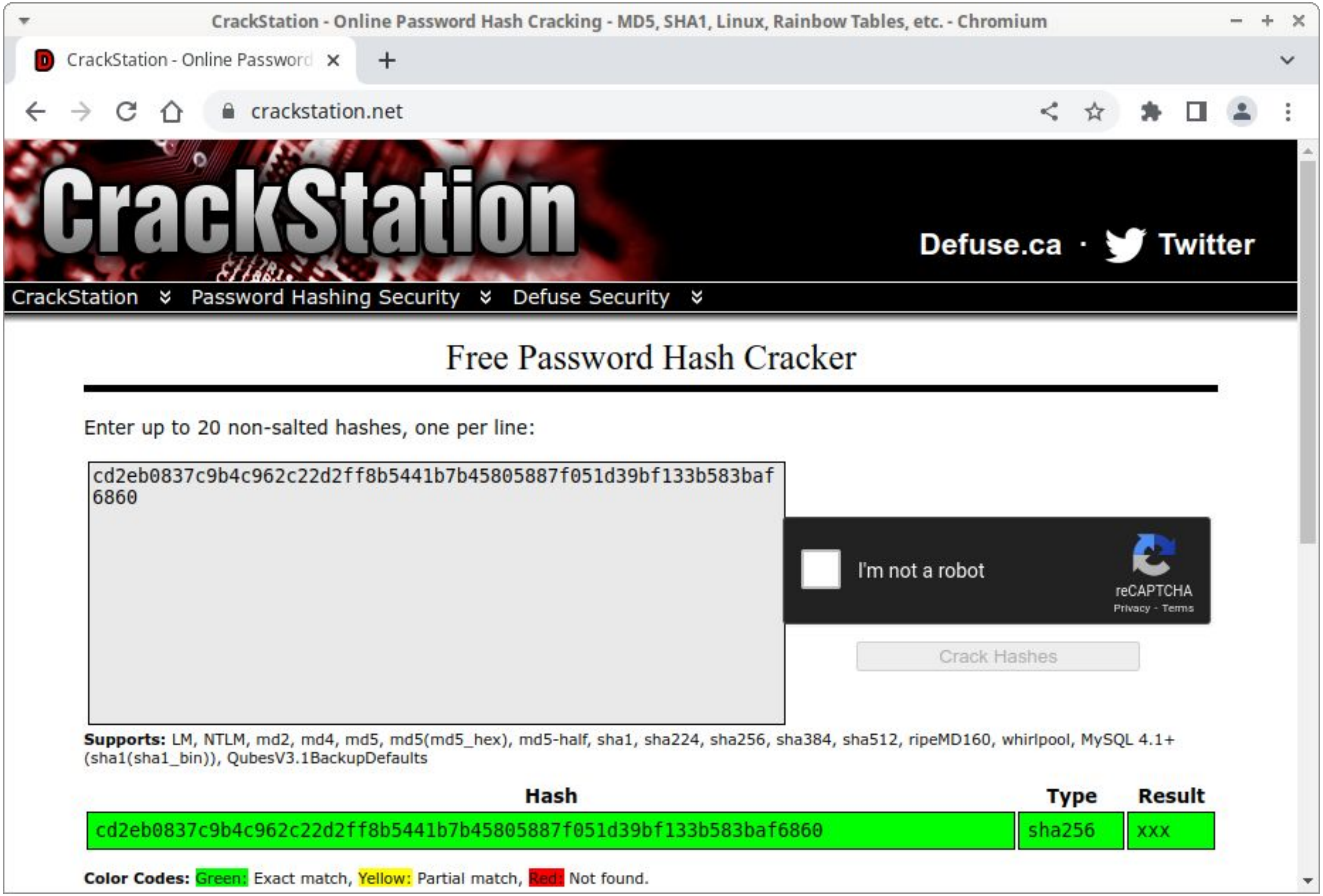
Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Download CrackStation's Wordlist

How CrackStation Works

CrackStation uses massive pre-computed lookup tables to crack password hashes. These tables store a mapping between the hash of a password, and the correct password for that hash. The hash values are indexed so that it is possible to quickly search the database for a given hash. If the hash is present in the database, the password can be recovered in a fraction of a second. This only works for "unsalted" hashes. For information on password hashing systems that are not vulnerable to pre-computed lookup tables, see our hashing security page.

Data Storage Attacks


CrackStation - Online Password Hash Cracking - MD5, SHA1, Linux, Rainbow Tables, etc. - Chromium
CrackStation - Online Password
crackstation.net
CrackStation
Defuse.ca · Twitter
CrackStation
Password Hashing Security
Defuse Security
I'm not a robot
reCAPTCHA
Privacy - Terms
Crack Hashes

Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

```
cd2eb0837c9b4c962c22d2ff8b5441b7b45805887f051d39bf133b583baf
6860
```

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
cd2eb0837c9b4c962c22d2ff8b5441b7b45805887f051d39bf133b583baf6860	sha256	xxx

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

Data Storage Attacks

- **Solution:**
  - Store hash codes of ***salted*** passwords
    - `hash_code = sha256('!@#$%^' + password)`
    - hash codes of salted passwords will not be found in a rainbow table

Data Storage Attacks

- **Problem:**
  - If an attacker learns the app’s salt string, then the attacker (with lots of effort) might generate a rainbow table that contains hash codes for common salted passwords
    - … And so might discover the app’s (common) passwords
- **Solution:**
  - Use a different salt string for each password

Data Storage Attacks

Salting and sha256 hashing in Python

```
$ python
>>> import werkzeug.security
>>> h = werkzeug.security.generate_password_hash('xxx', 'pbkdf2')
>>> h
'pbkdf2:sha256:600000$G8hNoAKf6ttD5iBa$262b04f2f287889ddffd77b0a735
b543491954d917d20bb36ae6ce2bd0ee5fde'
>>> werkzeug.security.check_password_hash(h, 'xxx')
True
>>> werkzeug.security.check_password_hash(h, 'yyy')
False
>>> quit()
$
```

Data Storage Attacks

Salting and sha256 hashing in Python

algorithm salt hashcode

```
pbkdf2:sha256:600000$G8hNoAKf6ttD5iBa$262b04
f2f287889ddffd77b0a735b543491954d917d20bb36a
e6ce2bd0ee5fde
```

Data Storage Attacks

- See **PennyAdmin8bSaltHash** app
  - runserver.py
  - **penny.sql**, penny.sqlite
  - database.py
  - header.html, footer.html
  - index.html, show.html,
  - add.html, delete.html, reportresults.html
  - login.html, signup.html, loggedout.html
  - top.py, penny.py, **auth.py**

Data Storage Attacks

- Q: Project concern?

- A: **Yes**
  - If your app does its own checking of user passwords

Agenda

- Data storage attacks
- **Data comm attacks**
- Third-party authentication (briefly)
  - CAS authentication
  - Microsoft EntraID authentication
  - Google authentication
  - Auth0 authentication

Data Comm Attacks

- **Problem:**
  - Attacker may access data during comm between PennyAdmin app and browser
- **Solution:**
  - ***Hypertext Transfer Protocol Secure (HTTPS)***

Data Comm Attacks

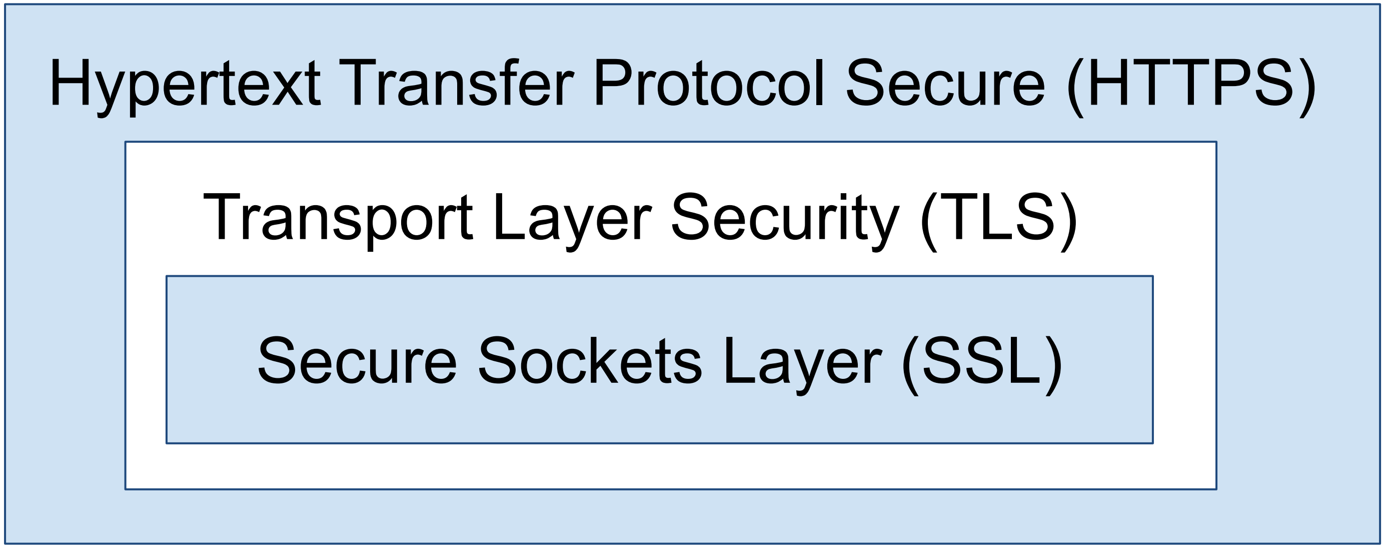
- **Technical** advantages of using HTTPS
  - Confidentiality
    - Prohibits ***message eavesdropping attacks***
  - Integrity
    - Prohibits ***message tampering attacks***
  - Authentication
    - Prohibits ***message forgery attacks***

Data Comm Attacks

- **Business** advantages of using HTTPS
  - Increases user confidence/trust in website
  - Increases Google search rank of website

Data Comm Attacks

- How HTTPS works:



Hypertext Transfer Protocol Secure (HTTPS)

Transport Layer Security (TLS)

Secure Sockets Layer (SSL)

Data Comm Attacks

- How to use HTTPS:
  - Configure server & app to use (& require use of) HTTPS
  - Command browser to send request specifying HTTPS as protocol
    - `https://…`

Data Comm Attacks

- How to configure server & app to use (& require use of) HTTPS:
  - Depends upon server…

Data Comm Attacks

- **Render server**
  - Already configured to use (& require use of) HTTPS
    - When server receives **`http`**`://`*`something`* request, it sends redirect for **`https`**`://`*`something`* request
  - So:
    - **Server**: Do nothing!
    - **App**: Do nothing!

Data Comm Attacks

- **Heroku server**
  - Already configured to use (but not require use of) HTTPS
    - When server receives **https:**//*something* request, it uses HTTPS
    - When server receives **http:**//*something* request, it uses HTTP
  - So
    - **Server**: (Regrettably) Do nothing!
    - **App**: Small change…

Data Comm Attacks

- **Solution 1:**
  - App explicitly performs redirects

Data Comm Attacks

- See **PennyAdmin9aHttps** app
  - runserver.py
  - penny.sql, penny.sqlite
  - database.py
  - header.html, footer.html
  - index.html, show.html,
  - add.html, delete.html, reportresults.html
  - login.html, signup.html, loggedout.html
  - top.py, **penny.py**, auth.py

Data Comm Attacks

- **Solution 2:**
  - ***`flask_talisman`*** module

Data Comm Attacks

- See **PennyAdmin9bHttps** app
  - runserver.py
  - penny.sql, penny.sqlite
  - database.py
  - header.html, footer.html
  - index.html, show.html,
  - add.html, delete.html, reportresults.html
  - login.html, signup.html, loggedout.html
  - top.py, **penny.py**, auth.py

Data Comm Attacks

- Notes:
  - Good to design your app to require use of HTTPS even when the app server already forces use of HTTPS
  - flask_talisman implements some additional security measures
  - Need not configure Flask test server to use (or require use of) HTTPS
    - But if you want to…
    - Or if you're using Google authentication…

Data Comm Attacks

- How to configure Flask test server & app to use (& require use of) HTTPS:

Data Comm Attacks

- **Preliminary step**: Get a ***certificate*** for your app

- **Option 1**: Get a certificate that is signed by a ***certificate authority***

Data Comm Attacks

Certificate authorities:

Rank	Authority	Market Share
1	Let’s Encrypt	59.9%
2	GlobalSign	21.0%
3	Sectigo	5.6%
4	GoDaddy	3.9%
5	DigiCert	3.1%

https://en.wikipedia.org/wiki/Certificate_authority#Providers
(as of July 2024, the most recent data available on Wikipedia as of Nov 2025)

Data Comm Attacks

- **Preliminary step**: Get a certificate for your app

- **Option 1**: Buy a certificate that is signed by a certificate authority
- **Option 2**: Create a ***self-signed certificate***

Data Comm Attacks

Linux, Mac, MS Windows Git Bash:

```
$ openssl req -x509 -newkey rsa:4096 -nodes -out cert.pem -keyout key.pem -days 365
Generating a RSA private key
..........................................................++++
.......++++
writing new private key to 'key.pem'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]: US
State or Province Name (full name) [Some-State]: NJ
Locality Name (eg, city) []: Princeton
Organization Name (eg, company) [Internet Widgits Pty Ltd]: Princeton University
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []: localhost
Email Address []:
$
```

Output: cert.pem, key.pem

Data Comm Attacks

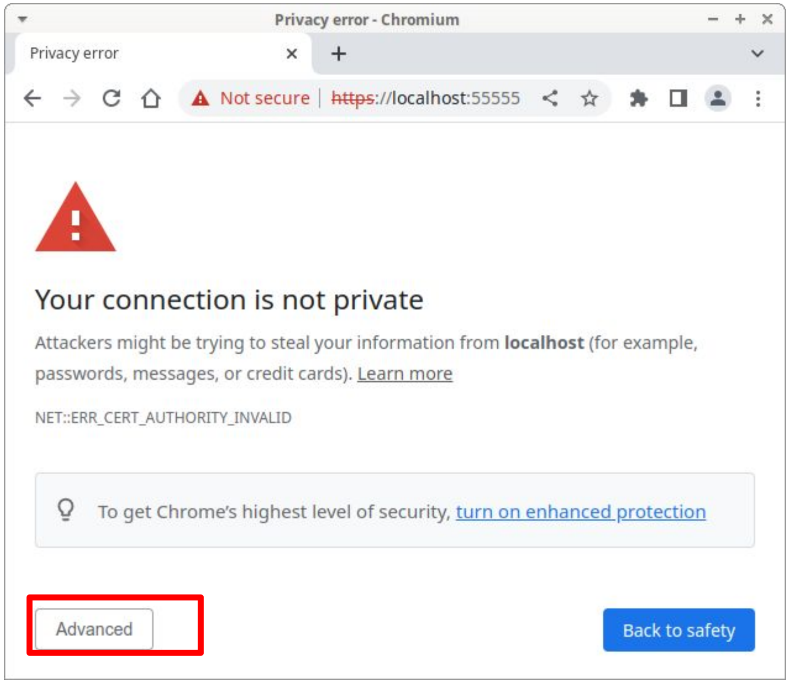
- Self-signed certificate
  - Confidentiality: yes
  - Integrity: yes
  - Authentication: no

Data Comm Attacks

- See **PennyAdmin09cHttpsLocal** app
  - **runserver.py**
  - penny.sql, penny.sqlite
  - database.py
  - header.html, footer.html
  - index.html, show.html,
  - add.html, delete.html, reportresults.html
  - login.html, signup.html, loggedout.html
  - top.py, penny.py, auth.py

Data Comm Attacks

- See **PennyAdmin09cHttpsLocal** app
  - On local computer with Flask test server (using self-signed certif)



Privacy error - Chromium

Privacy error

Not secure | ~~https~~://localhost:55555

Your connection is not private

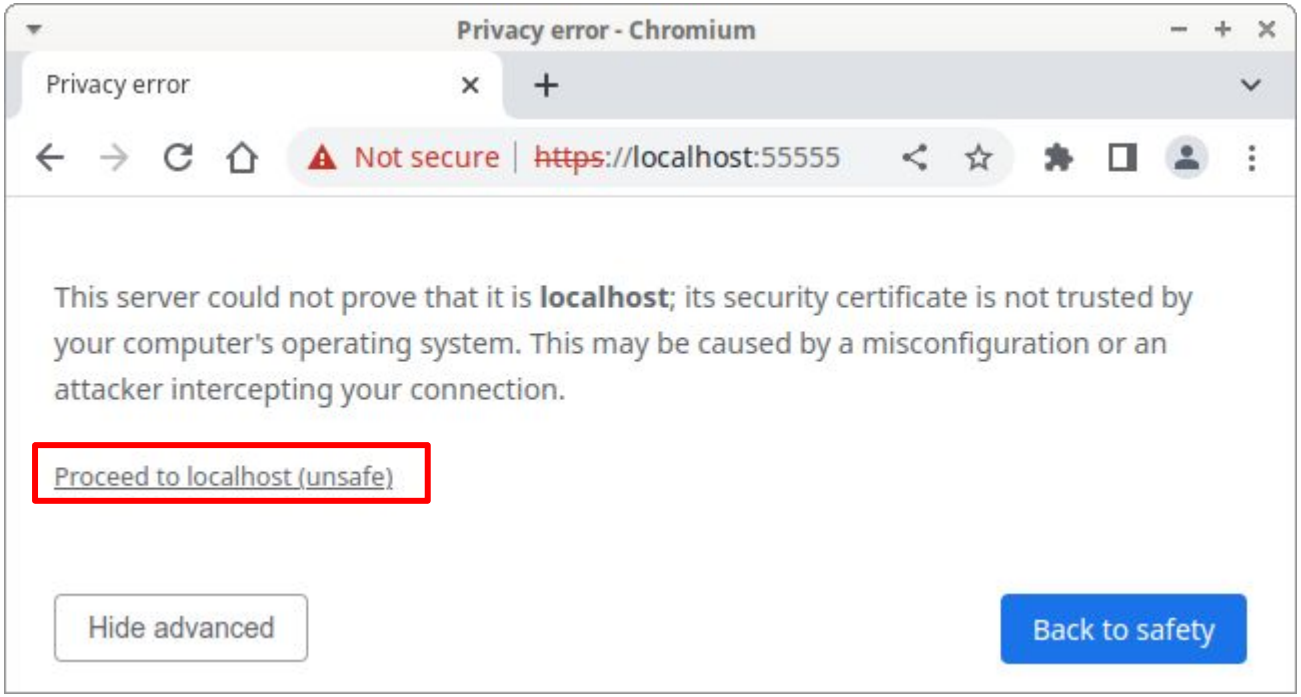
Attackers might be trying to steal your information from **localhost** (for example, passwords, messages, or credit cards). Learn more

NET::ERR_CERT_AUTHORITY_INVALID

To get Chrome's highest level of security, turn on enhanced protection

Data Comm Attacks

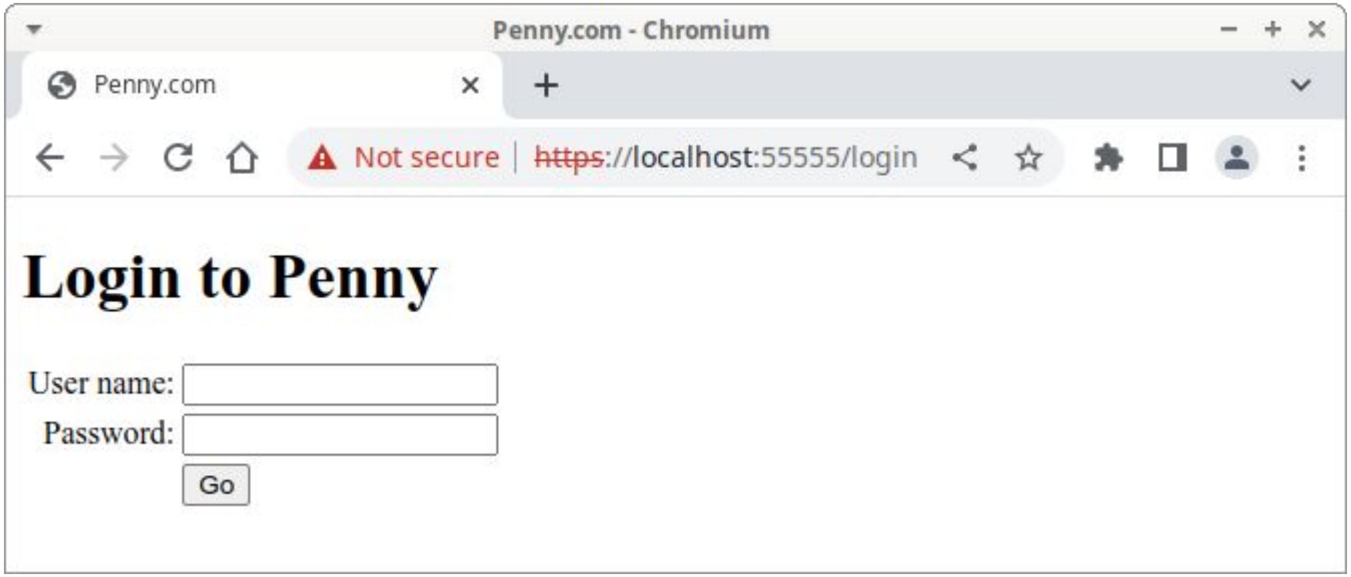
- See **PennyAdmin09cHttpsLocal** app
  - On local computer with Flask test server (using self-signed certif)



This server could not prove that it is **localhost**; its security certificate is not trusted by your computer's operating system. This may be caused by a misconfiguration or an attacker intercepting your connection.

Data Comm Attacks

- See **PennyAdmin09cHttpsLocal** app
  - On local computer with Flask test server (using self-signed certif)



Penny.com - Chromium

Penny.com

Not secure | ~~https~~://localhost:55555/login

Login to Penny

User name:

Password:

Data Comm Attacks

- Q: Project concern?
- A: **Yes**

Agenda

- Data storage Attacks
- Data comm attacks
- **Third-party authentication (briefly)**
  - CAS authentication
  - Microsoft EntraID authentication
  - Google authentication
  - Auth0 authentication

Agenda

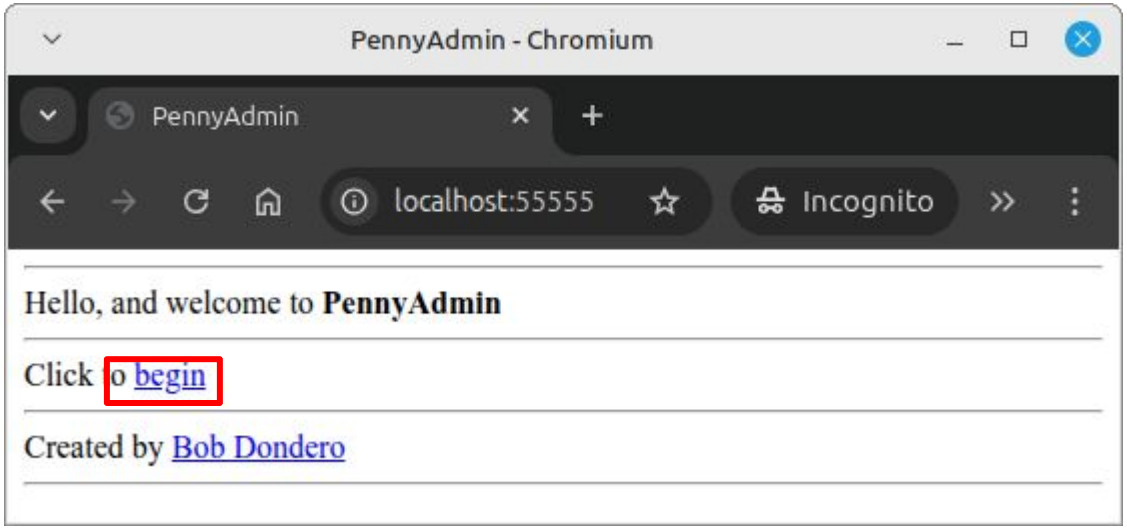
- Data storage attacks
- Data comm attacks
- Third-party authentication (briefly)
  - **CAS authentication**
  - Microsoft EntraID authentication
  - Google authentication
  - Auth0 authentication

CAS Authentication

- See **PennyAdminCas** app
  - Its behavior…

CAS Authentication

- See **PennyAdminCas** app (cont.)



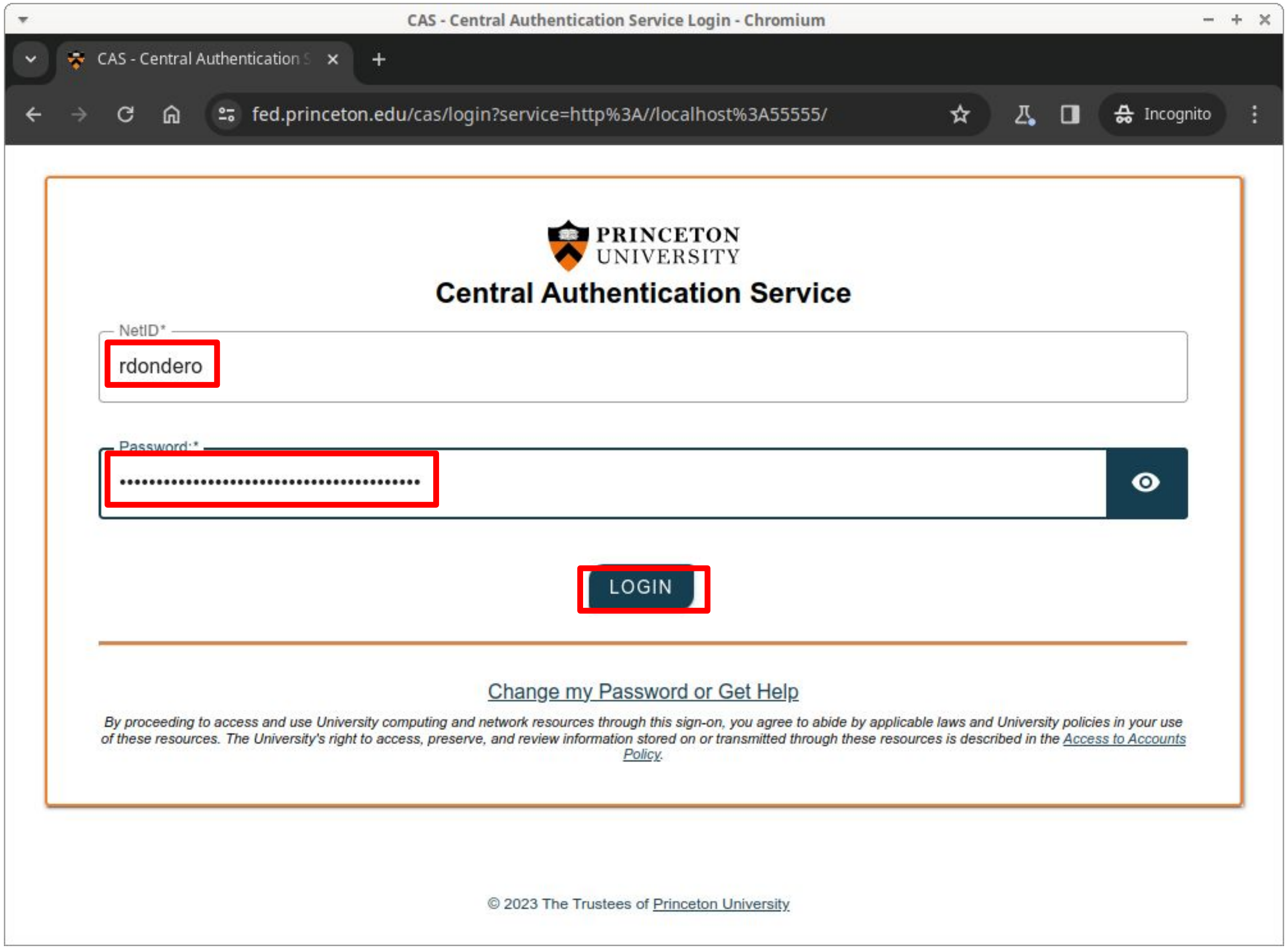
Hello, and welcome to **PennyAdmin**

Click to begin

Created by Bob Dondero

CAS Authentication

- See **PennyAdminCas** app (cont.)



CAS - Central Authentication Service Login - Chromium

CAS - Central Authentication S

fed.princeton.edu/cas/login?service=http%3A//localhost%3A55555/

Incognito

PRINCETON UNIVERSITY

Central Authentication Service

NetID*: rdondero

Password:*: ••

LOGIN

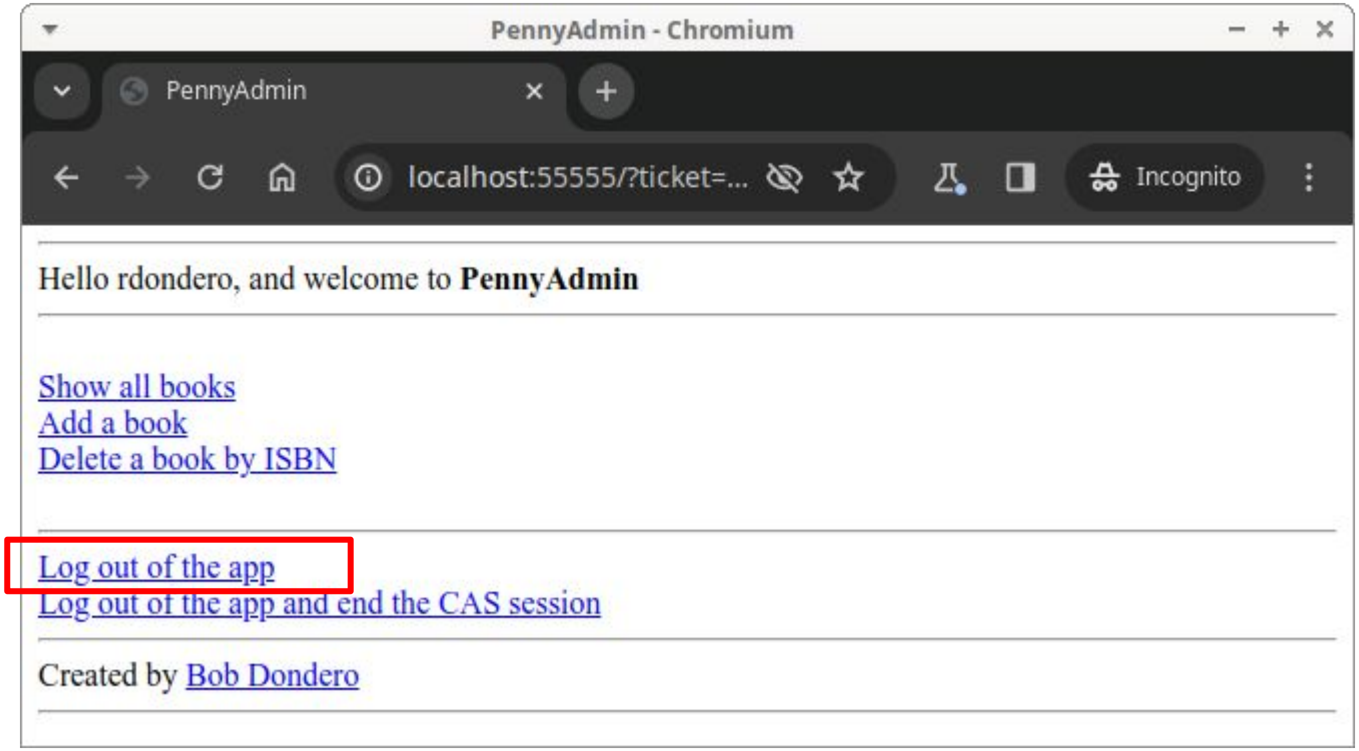
Change my Password or Get Help

By proceeding to access and use University computing and network resources through this sign-on, you agree to abide by applicable laws and University policies in your use of these resources. The University's right to access, preserve, and review information stored on or transmitted through these resources is described in the Access to Accounts Policy.

© 2023 The Trustees of Princeton University

CAS Authentication

- See **PennyAdminCas** app (cont.)



Hello rdondero, and welcome to **PennyAdmin**

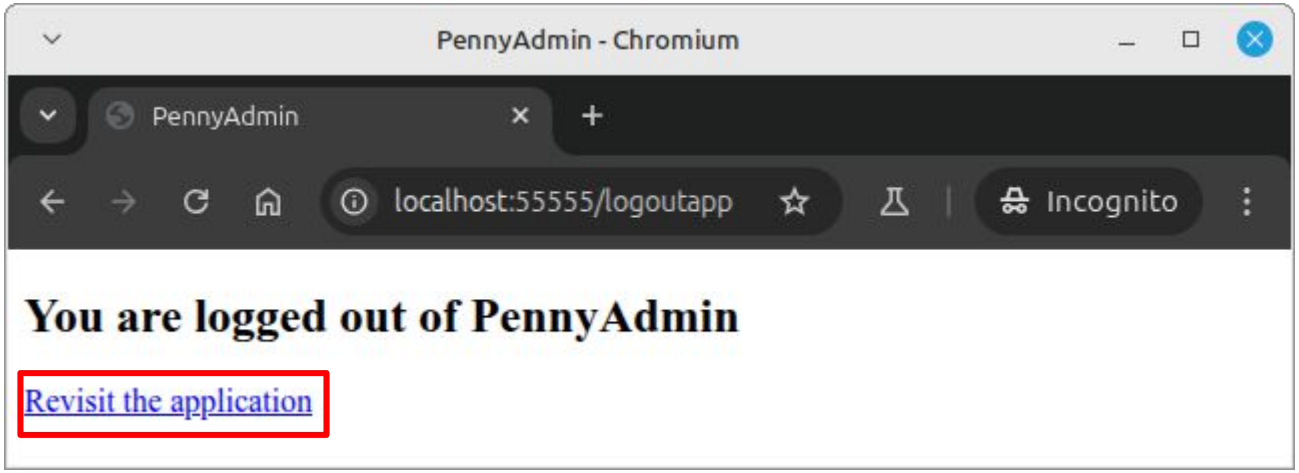
Show all books
Add a book
Delete a book by ISBN

Log out of the app
Log out of the app and end the CAS session

Created by Bob Dondero

CAS Authentication

- See **PennyAdminCas** app (cont.)

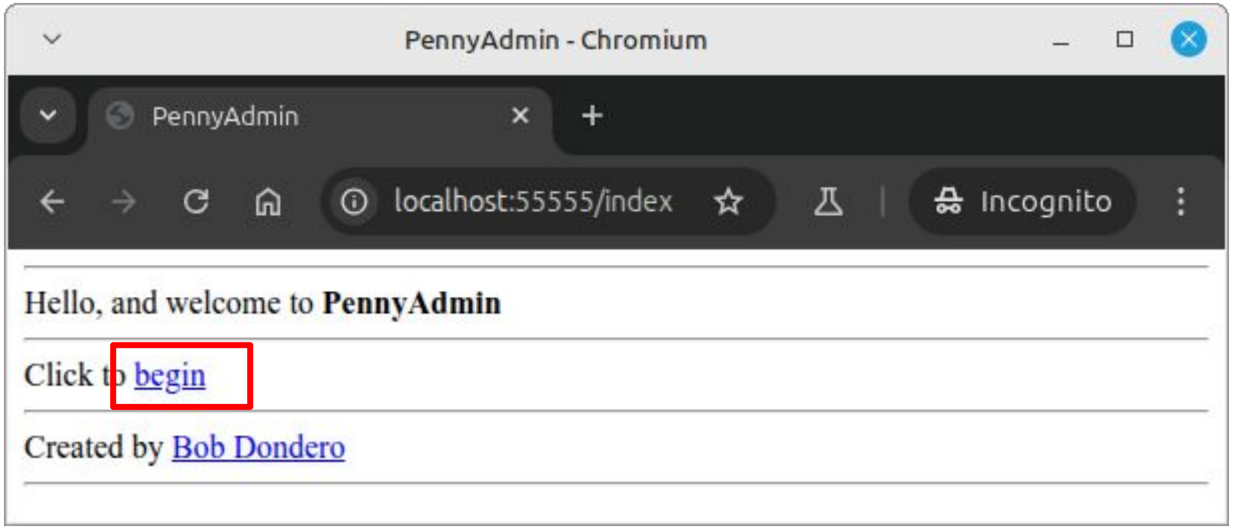


You are logged out of PennyAdmin

Revisit the application

CAS Authentication

- See **PennyAdminCas** app (cont.)



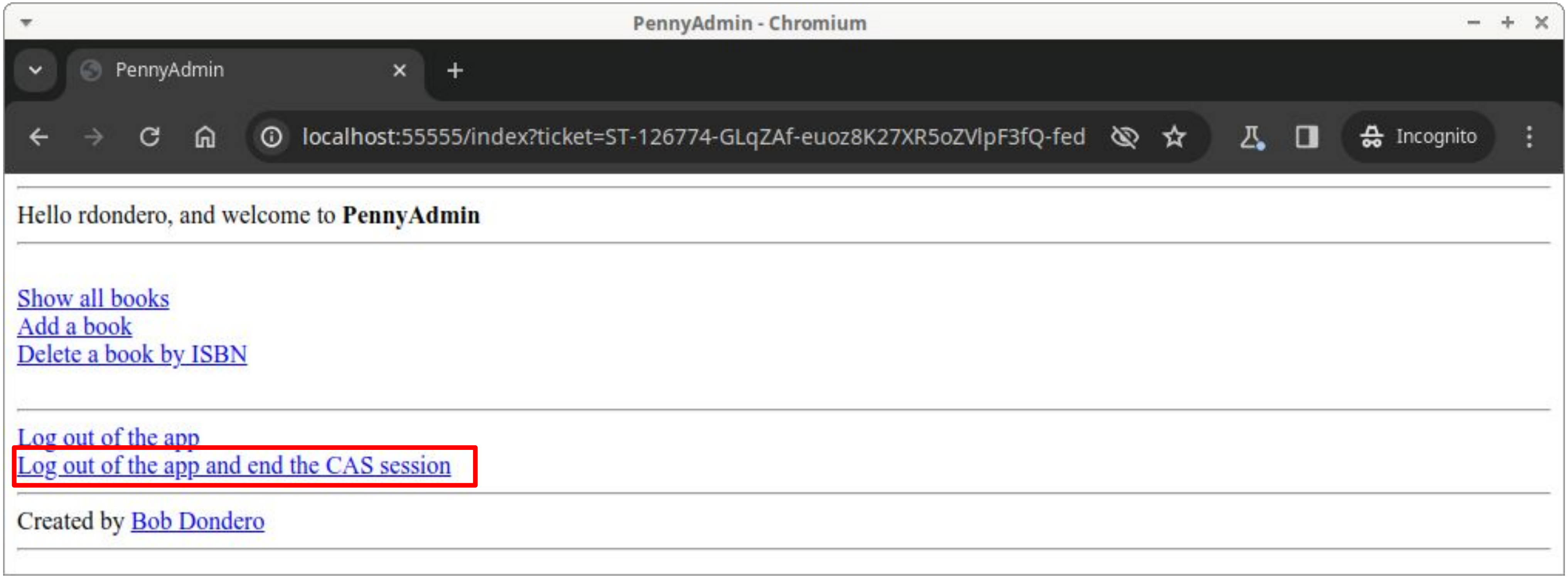
Hello, and welcome to **PennyAdmin**

Click to begin

Created by Bob Dondero

CAS Authentication

- See **PennyAdminCas** app (cont.)



PennyAdmin - Chromium
PennyAdmin
localhost:55555/index?ticket=ST-126774-GLqZAf-euoz8K27XR5oZVlpF3fQ-fed
Incognito

Hello rdondero, and welcome to **PennyAdmin**

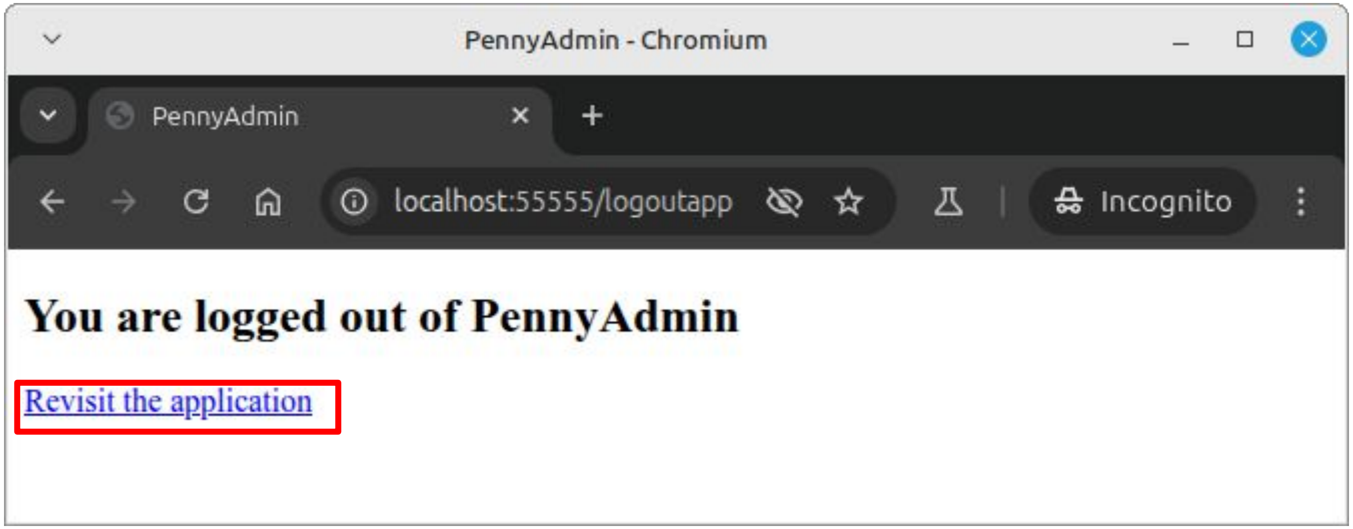
Show all books
Add a book
Delete a book by ISBN

Log out of the app
Log out of the app and end the CAS session

Created by Bob Dondero

CAS Authentication

- See **PennyAdminCas** app (cont.)

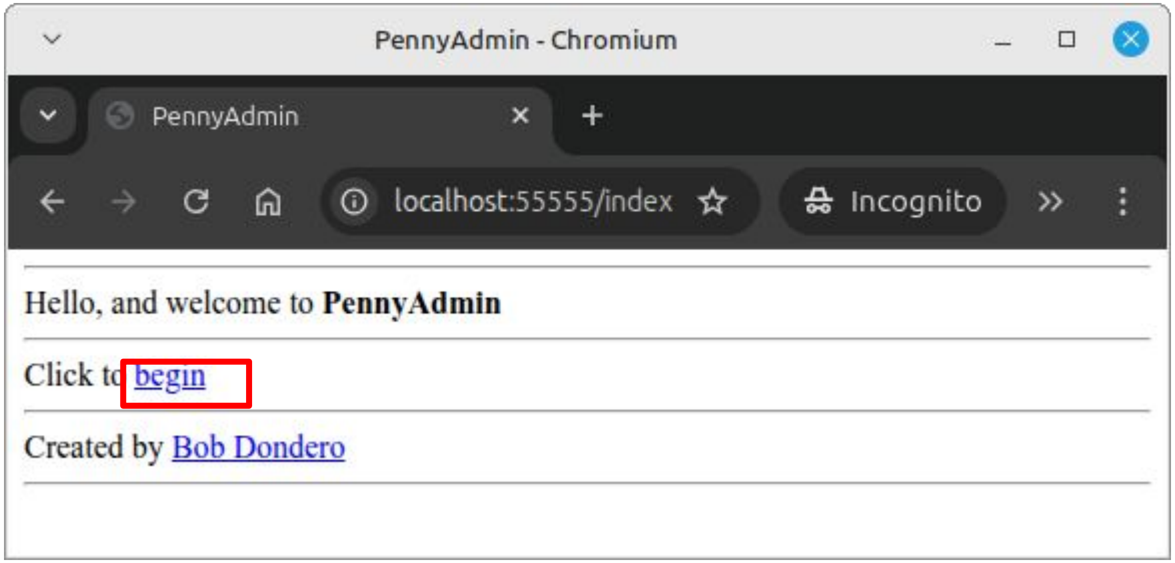


You are logged out of PennyAdmin

Revisit the application

CAS Authentication

- See **PennyAdminCas** app (cont.)



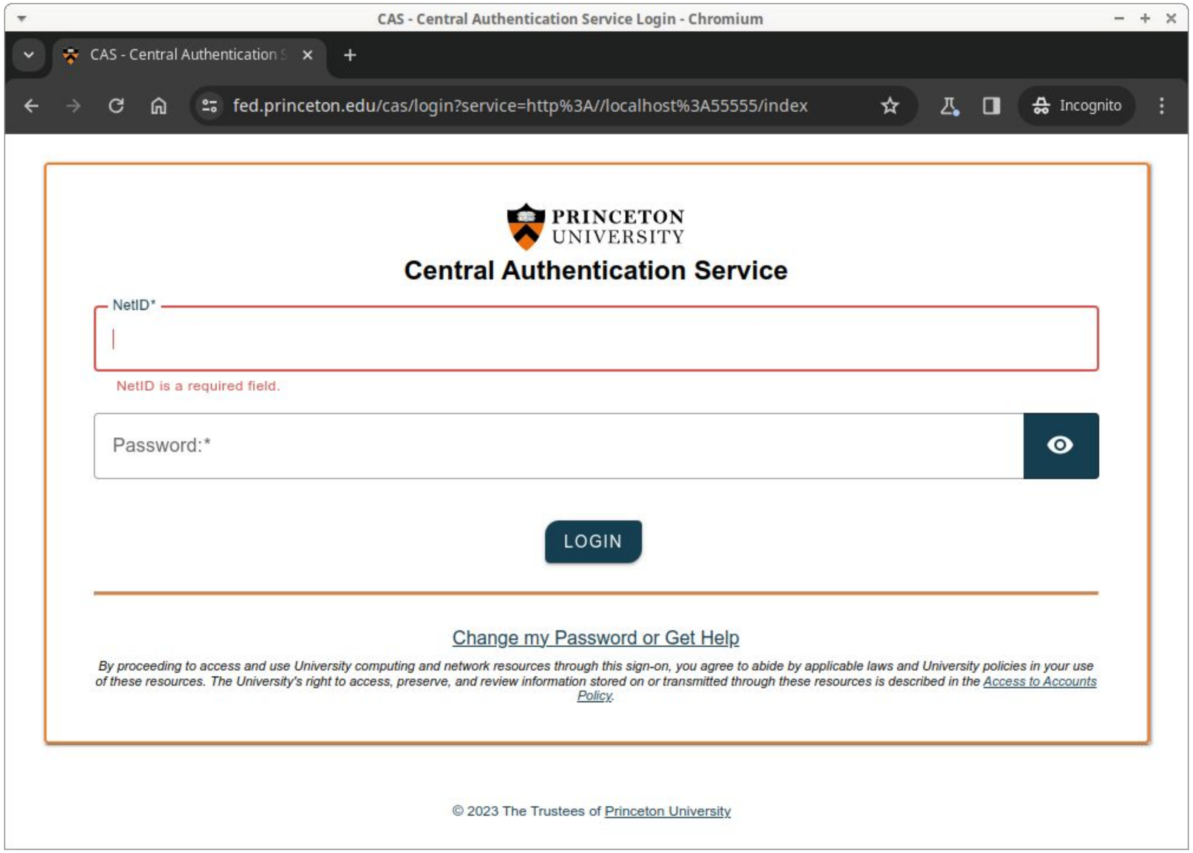
Hello, and welcome to **PennyAdmin**

Click to begin

Created by Bob Dondero

CAS Authentication

- See **PennyAdminCas** app (cont.)


CAS - Central Authentication Service Login - Chromium
CAS - Central Authentication S
Incognito
PRINCETON UNIVERSITY
LOGIN

fed.princeton.edu/cas/login?service=http%3A//localhost%3A55555/index

Central Authentication Service

NetID*

NetID is a required field.

Password:*

Change my Password or Get Help

By proceeding to access and use University computing and network resources through this sign-on, you agree to abide by applicable laws and University policies in your use of these resources. The University's right to access, preserve, and review information stored on or transmitted through these resources is described in the Access to Accounts Policy.

© 2023 The Trustees of Princeton University

CAS Authentication

- See **PennyAdminCas** app (cont.)
  - See optional lecture material for:
    - The code
    - How to run it on your local computer
    - How to run it on Render
    - How it works

CAS Authentication

- **Pros**
  - Application need not manage usernames or passwords
  - Application ***cannot*** access passwords!
  - Application is constrained to one user community
  - Princeton OIT will support for the foreseeable future

CAS Authentication

- **Cons**
  - Complex
  - Adds overhead, but only during user’s first visit to the app per browser session
  - Application is constrained to one user community!
  - Princeton OIT is phasing out
  - Deployment requires (minimal) Princeton OIT intervention

Agenda

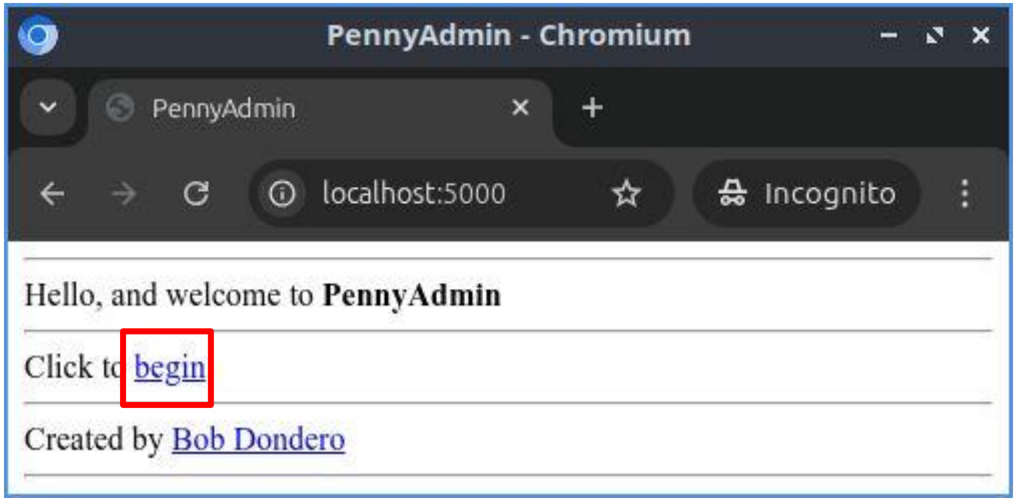
- Data comm attacks
- Third-party authentication (briefly)
  - CAS authentication
  - **Microsoft EntraID authentication**
  - Google authentication
  - Auth0 authentication

EntraID Authentication

- See **PennyAdminEntraID** app
  - Its behavior…

EntraID Authentication

- See **PennyAdminEntraID** app (cont.)



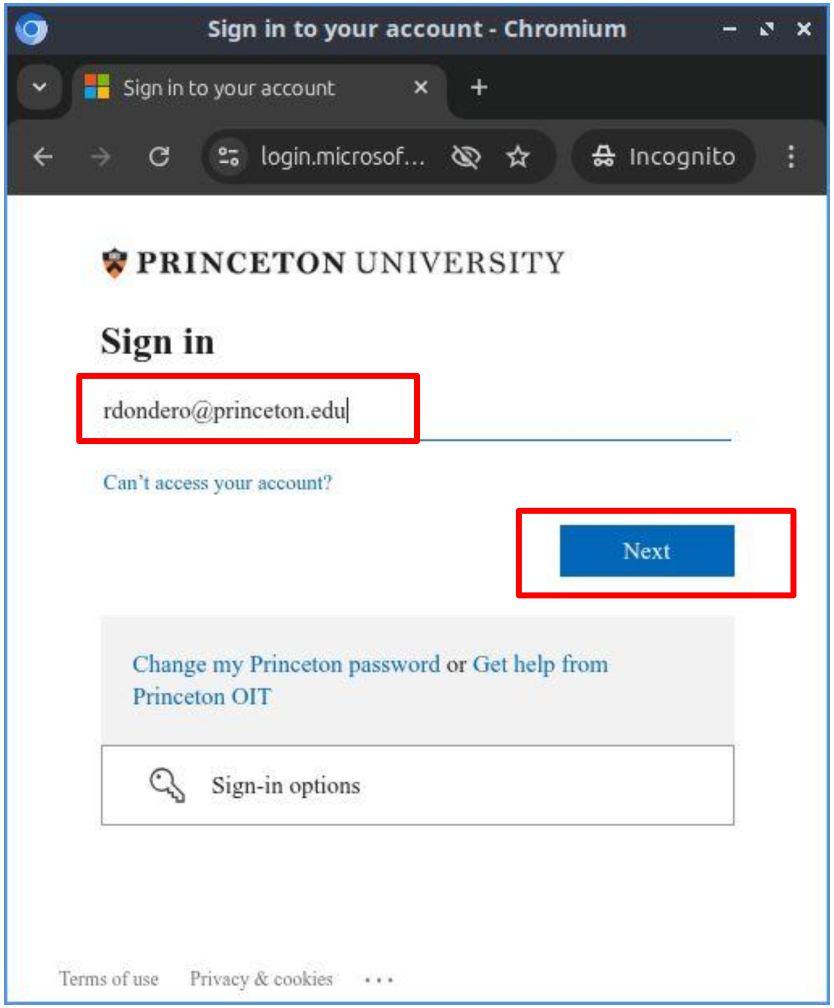
Hello, and welcome to **PennyAdmin**

Click to begin

Created by Bob Dondero

EntraID Authentication

- See **PennyAdminEntraID** app (cont.)



PRINCETON UNIVERSITY

Sign in

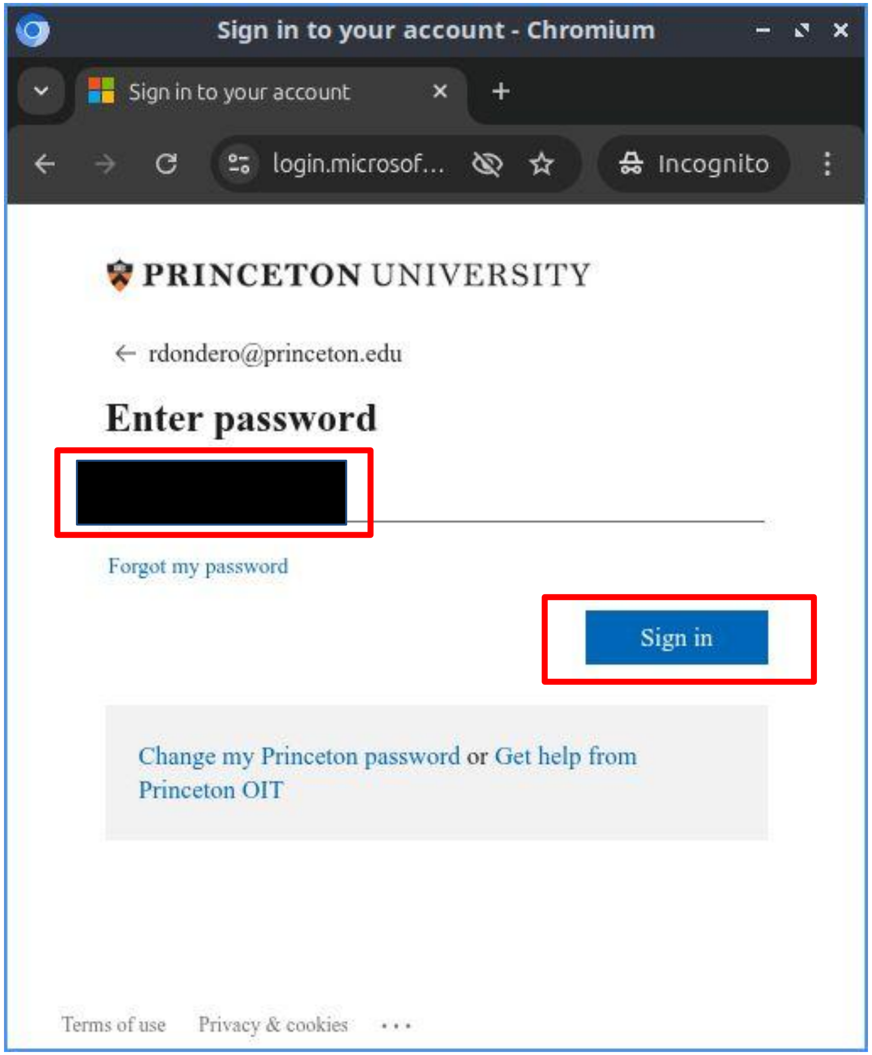
rdondero@princeton.edu

Can't access your account?

Change my Princeton password or Get help from Princeton OIT

EntralD Authentication

- See **<u>PennyAdminEntralD</u>** app (cont.)



Sign in to your account - Chromium

Sign in to your account

login.microsof...

Incognito

PRINCETON UNIVERSITY

← rdondero@princeton.edu

Enter password

[redacted]

Forgot my password

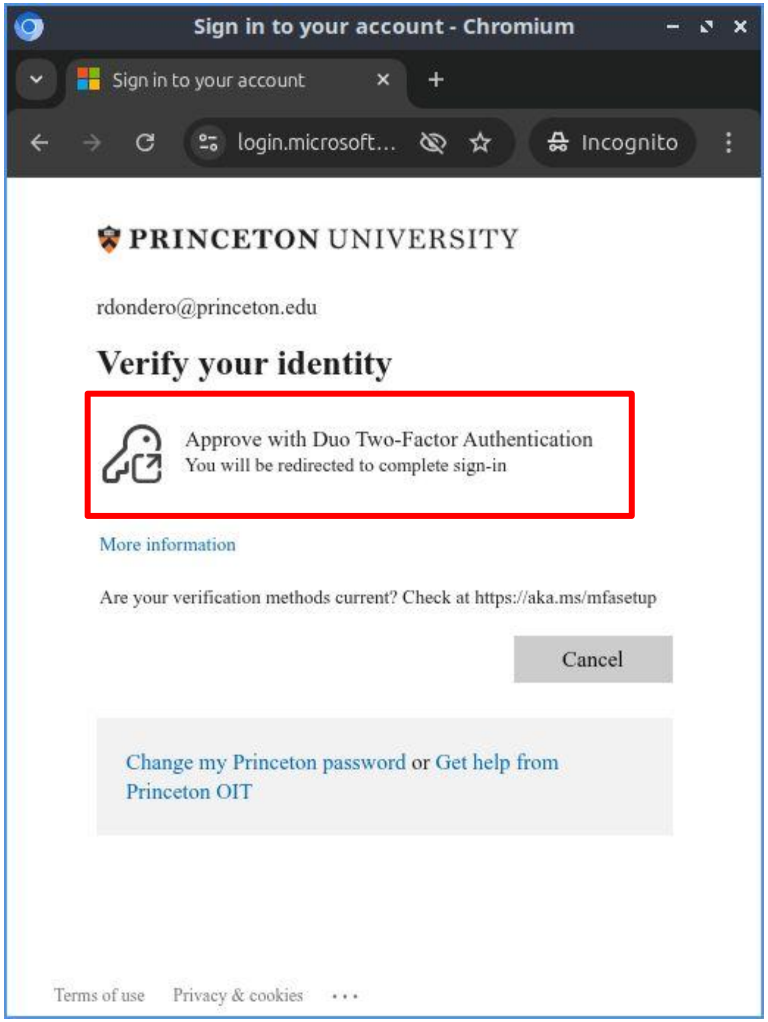
Sign in

Change my Princeton password or Get help from Princeton OIT

Terms of use Privacy & cookies ...

EntraID Authentication

- See **PennyAdminEntraID** app (cont.)



PRINCETON UNIVERSITY

rdondero@princeton.edu

Verify your identity

Approve with Duo Two-Factor Authentication
You will be redirected to complete sign-in

More information

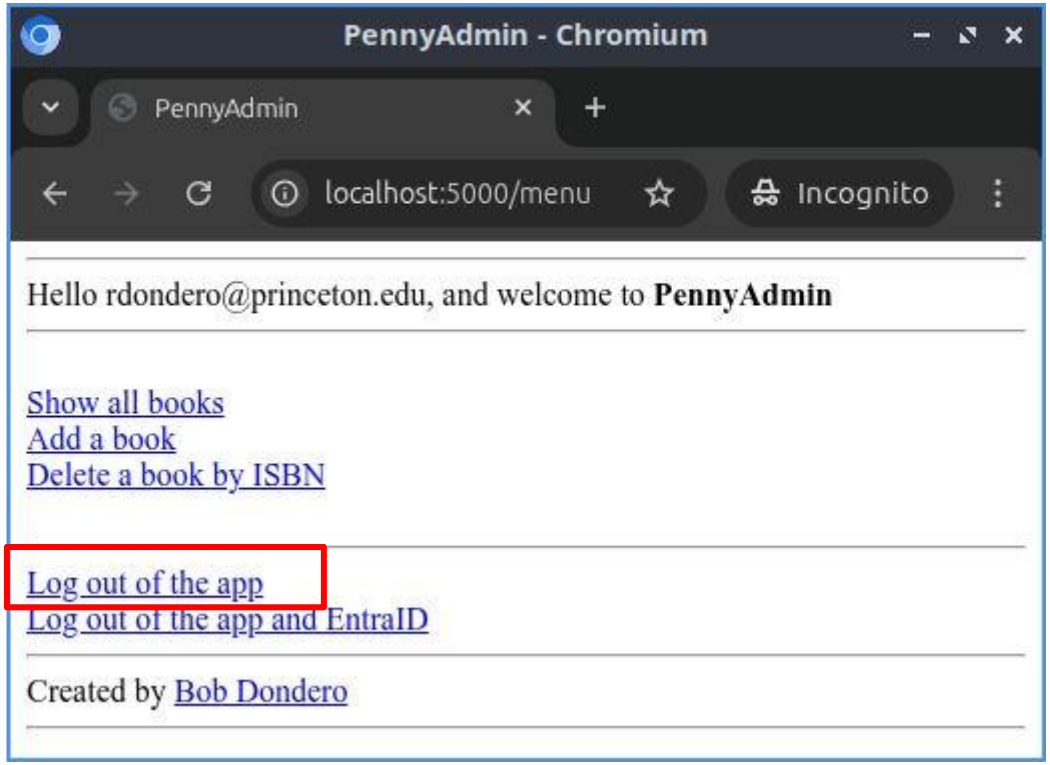
Are your verification methods current? Check at https://aka.ms/mfasetup

Cancel

Change my Princeton password or Get help from Princeton OIT

EntraID Authentication

- See **PennyAdminEntraID** app (cont.)



Hello rdondero@princeton.edu, and welcome to **PennyAdmin**

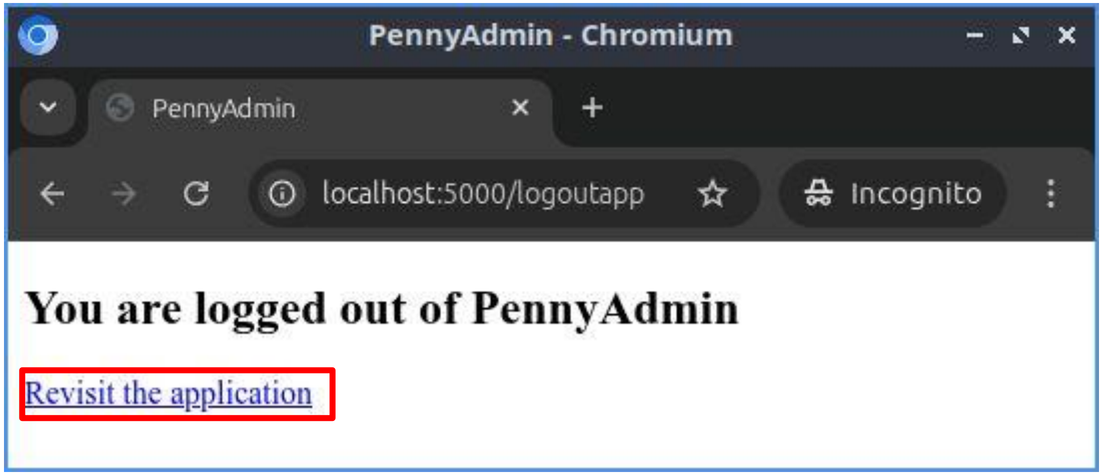
Show all books
Add a book
Delete a book by ISBN

Log out of the app
Log out of the app and EntraID

Created by Bob Dondero

EntraID Authentication

- See **PennyAdminEntraID** app (cont.)

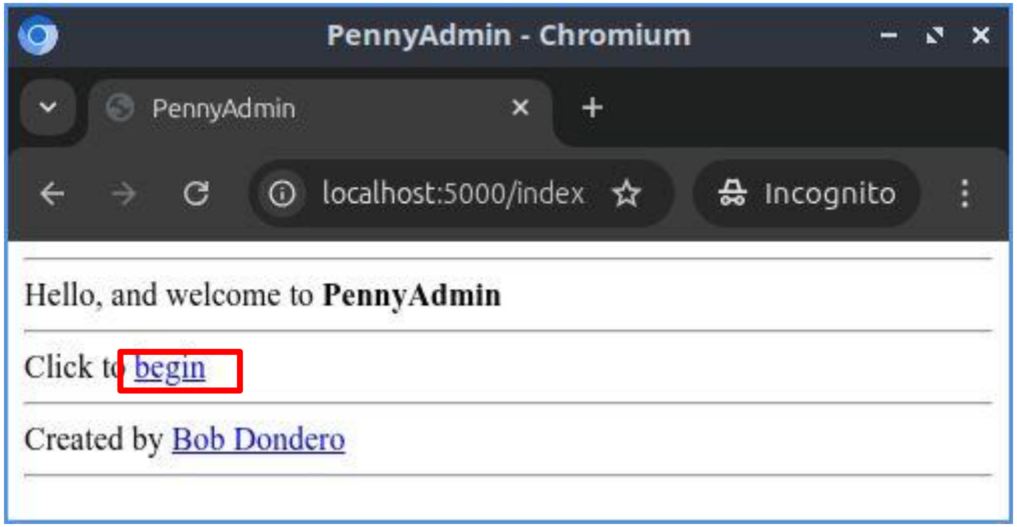


You are logged out of PennyAdmin

Revisit the application

EntraID Authentication

- See **PennyAdminEntraID** app (cont.)



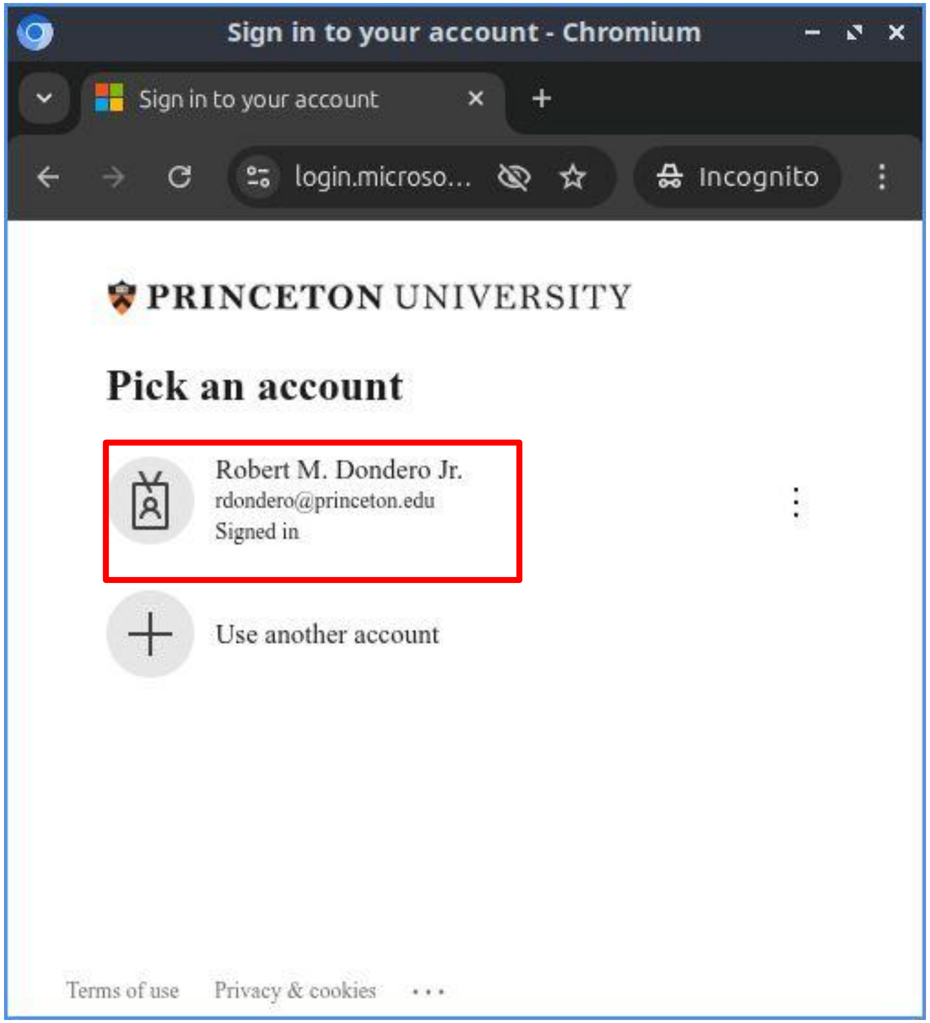
Hello, and welcome to **PennyAdmin**

Click to begin

Created by Bob Dondero

EntraID Authentication

- See **PennyAdminEntraID** app (cont.)



PRINCETON UNIVERSITY

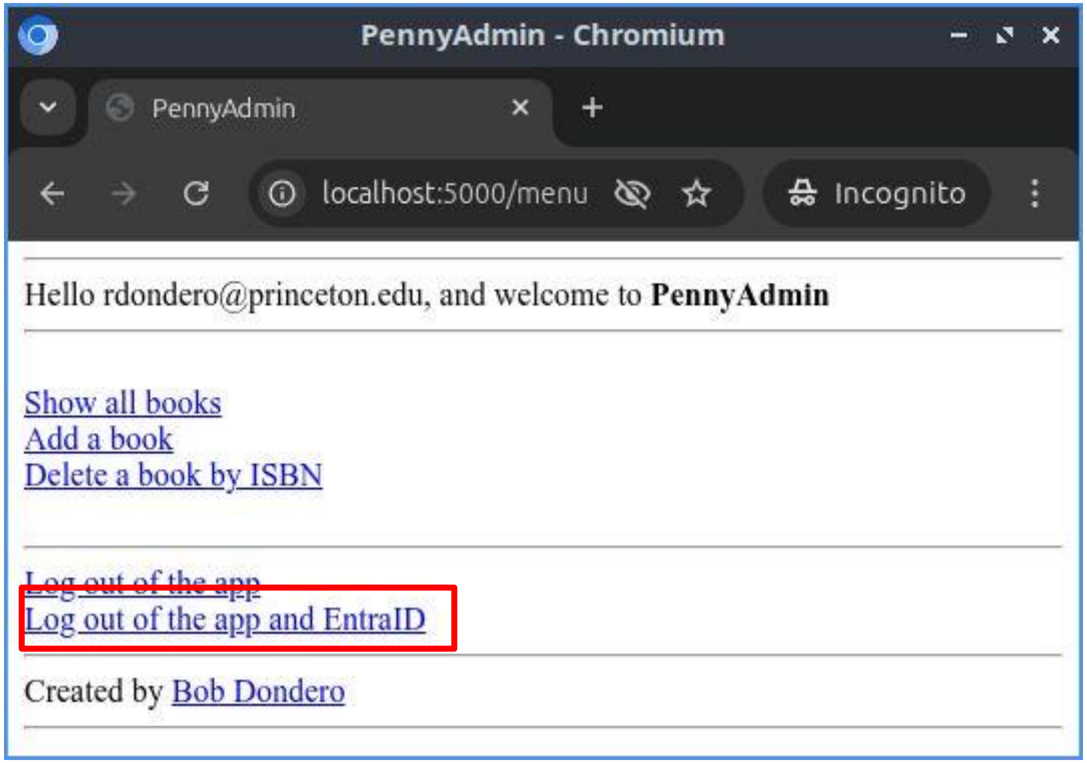
Pick an account

Robert M. Dondero Jr.
rdondero@princeton.edu
Signed in

Use another account

EntraID Authentication

- See **PennyAdminEntraID** app (cont.)



Hello rdondero@princeton.edu, and welcome to **PennyAdmin**

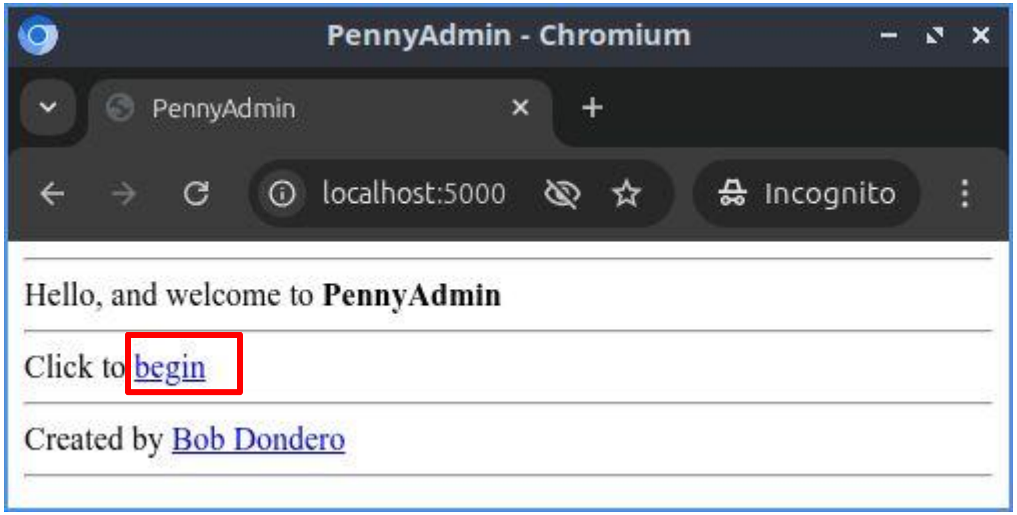
Show all books
Add a book
Delete a book by ISBN

Log out of the app
Log out of the app and EntraID

Created by Bob Dondero

EntraID Authentication

- See **PennyAdminEntraID** app (cont.)



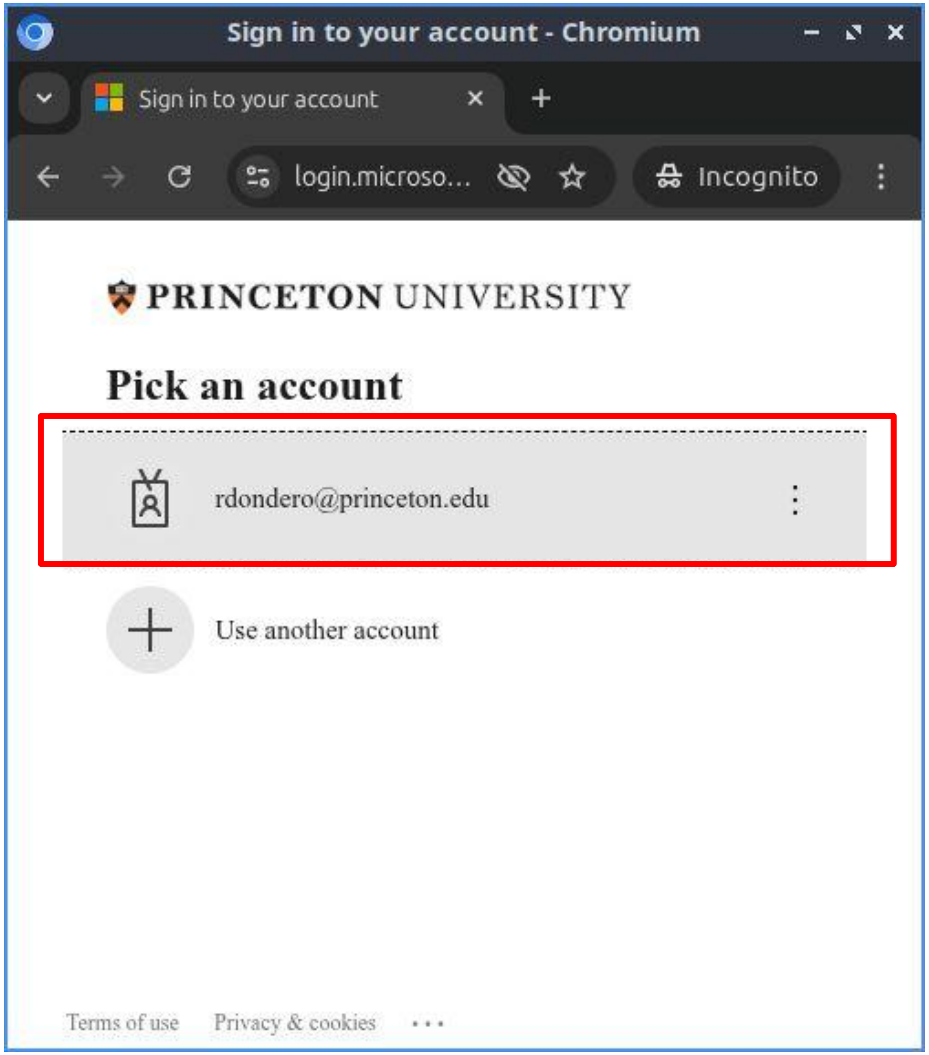
Hello, and welcome to **PennyAdmin**

Click to begin

Created by Bob Dondero

EntraID Authentication

- See **PennyAdminEntraID** app (cont.)



Sign in to your account - Chromium

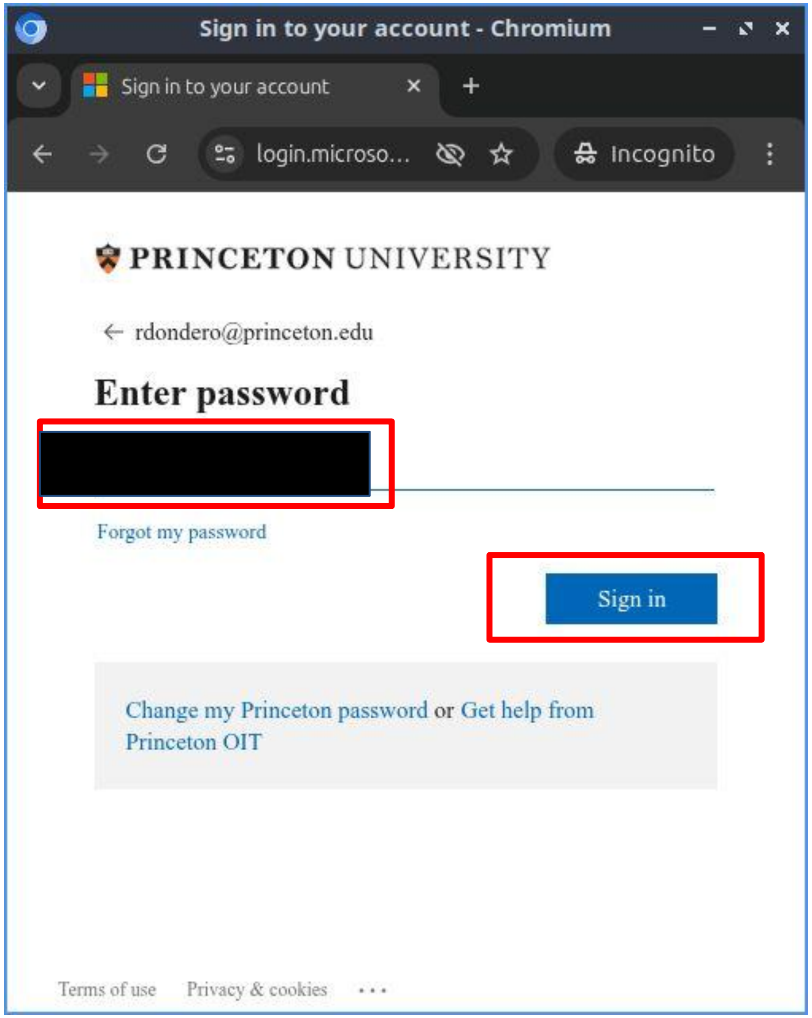
Pick an account

rdondero@princeton.edu

Use another account

EntraID Authentication

- See **PennyAdminEntraID** app (cont.)



Sign in to your account - Chromium

Sign in to your account

login.microso...

Incognito

PRINCETON UNIVERSITY

← rdondero@princeton.edu

Enter password

[redacted]

Forgot my password

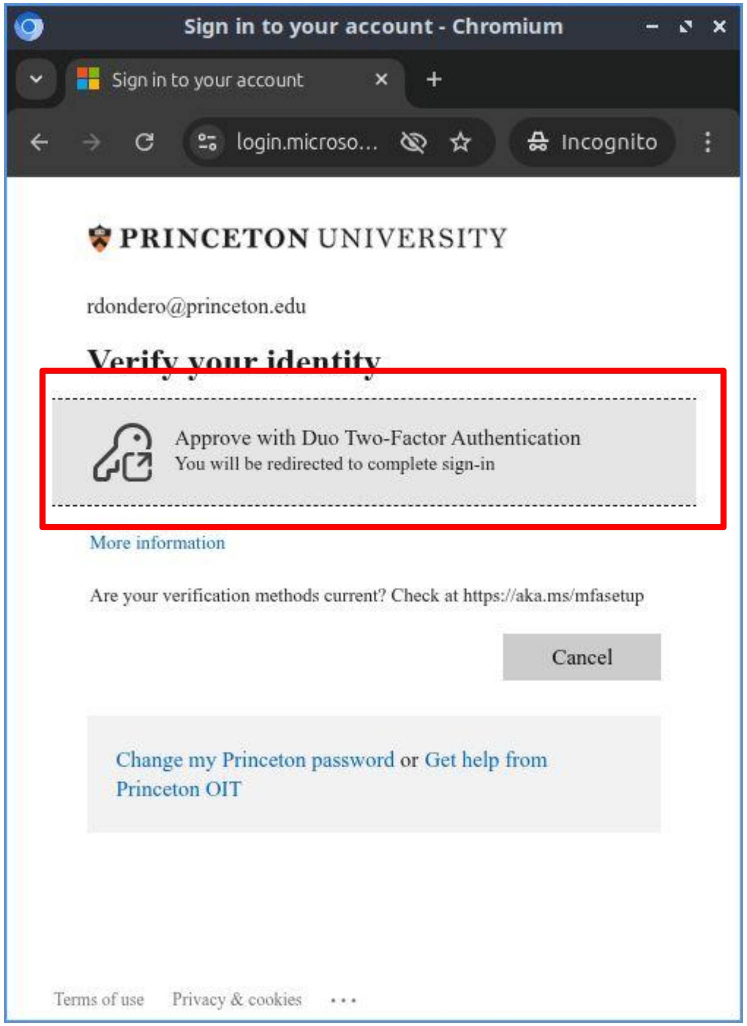
Sign in

Change my Princeton password or Get help from Princeton OIT

Terms of use Privacy & cookies ...

EntraID Authentication

- See **PennyAdminEntraID** app (cont.)



PRINCETON UNIVERSITY

rdondero@princeton.edu

Verify your identity

Approve with Duo Two-Factor Authentication
You will be redirected to complete sign-in

More information

Are your verification methods current? Check at https://aka.ms/mfasetup

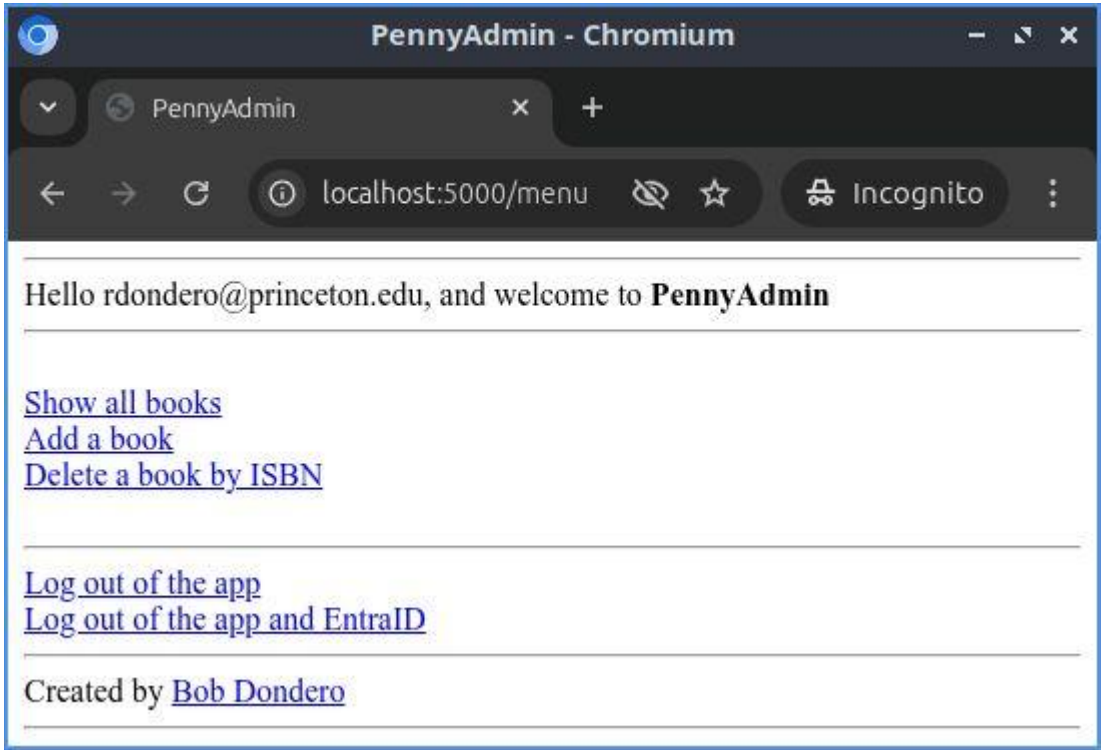
Cancel

Change my Princeton password or Get help from Princeton OIT

Terms of use Privacy & cookies ...

EntraID Authentication

- See **PennyAdminEntraID** app (cont.)



Hello rdondero@princeton.edu, and welcome to **PennyAdmin**

Show all books
Add a book
Delete a book by ISBN

Log out of the app
Log out of the app and EntraID

Created by Bob Dondero

EntraID Authentication

- See **PennyAdminEntraID** app (cont.)
  - See optional lecture material for:
    - The code
    - How to run it on your local computer
    - How to run it on Render

EntraID Authentication

- **Pros**
  - Application need not manage usernames or passwords
  - Application ***cannot*** access passwords!
  - Application is constrained to one user community
  - More standards-based than CAS
  - Princeton OIT is phasing in
  - No need for Princeton OIT intervention

EntraID Authentication

- **Cons**
  - Complex
  - Adds overhead, but only during user’s first visit to the app per browser session
  - Application is constrained to one user community!

Agenda

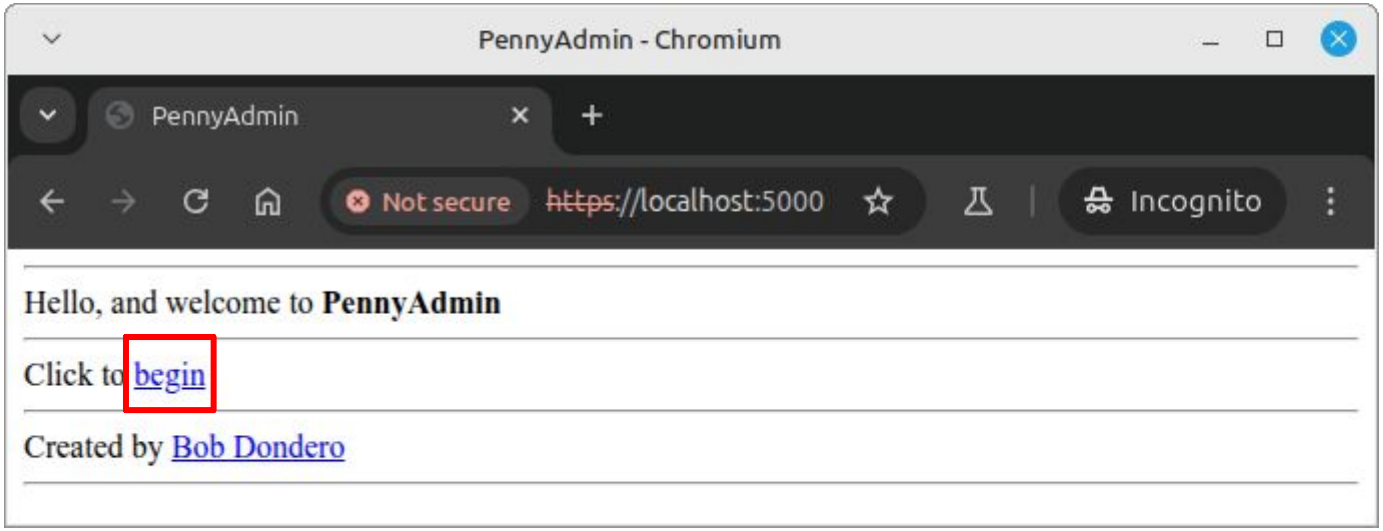
- Data comm attacks
- Third-party authentication (briefly)
  - CAS
  - Microsoft EntraID authentication
  - **Google authentication**
  - Auth0 authentication

Google Authentication

- See **PennyAdminGoogle** app
  - Its behavior…

Google Authentication

- See **PennyAdminGoogle** app (cont.)



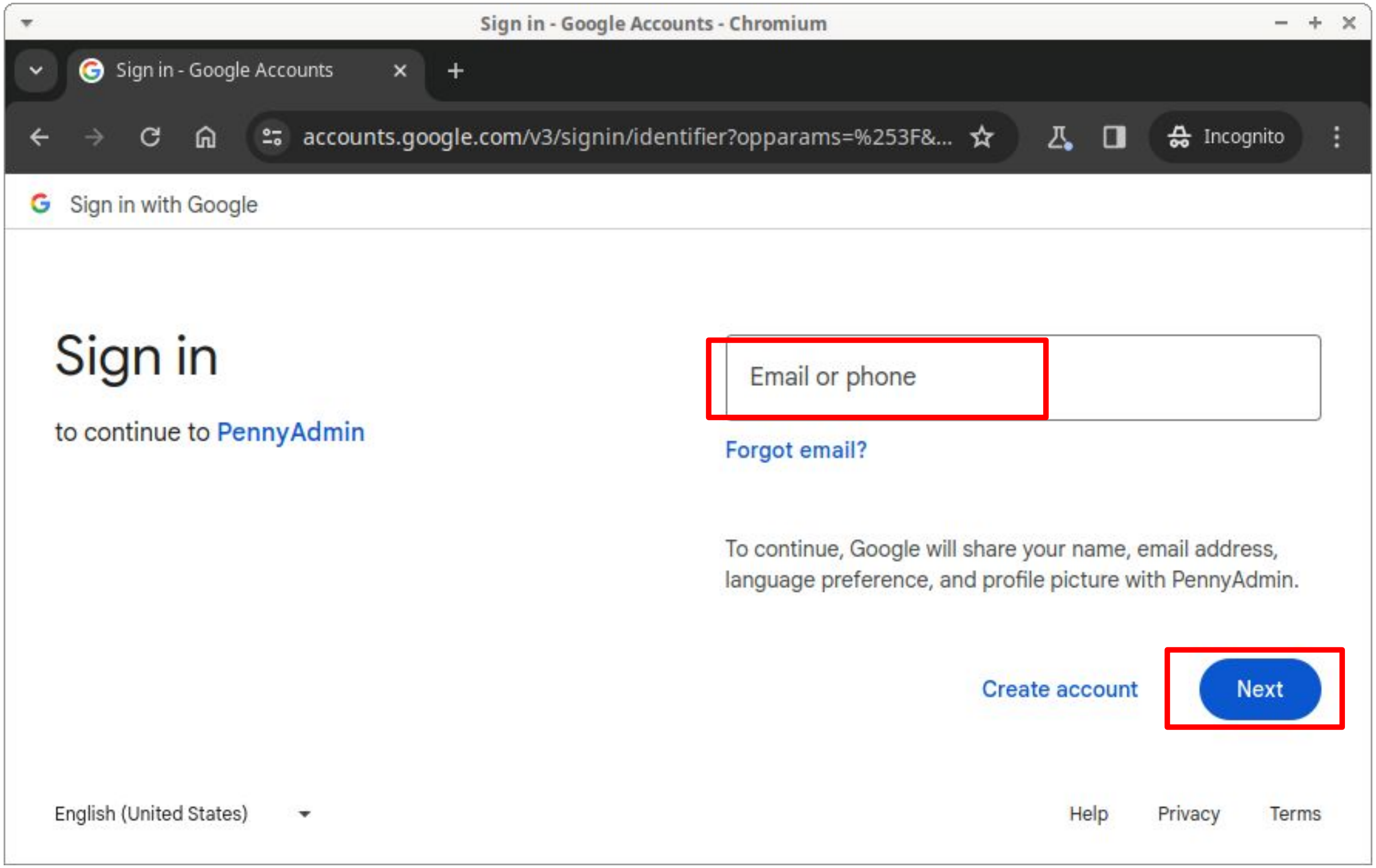
Hello, and welcome to **PennyAdmin**

Click to begin

Created by Bob Dondero

Google Authentication

- See **PennyAdminGoogle** app (cont.)



Sign in - Google Accounts - Chromium

Sign in - Google Accounts

accounts.google.com/v3/signin/identifier?opparams=%253F&...

Incognito

Sign in with Google

Sign in

to continue to PennyAdmin

Email or phone

Forgot email?

To continue, Google will share your name, email address, language preference, and profile picture with PennyAdmin.

Create account

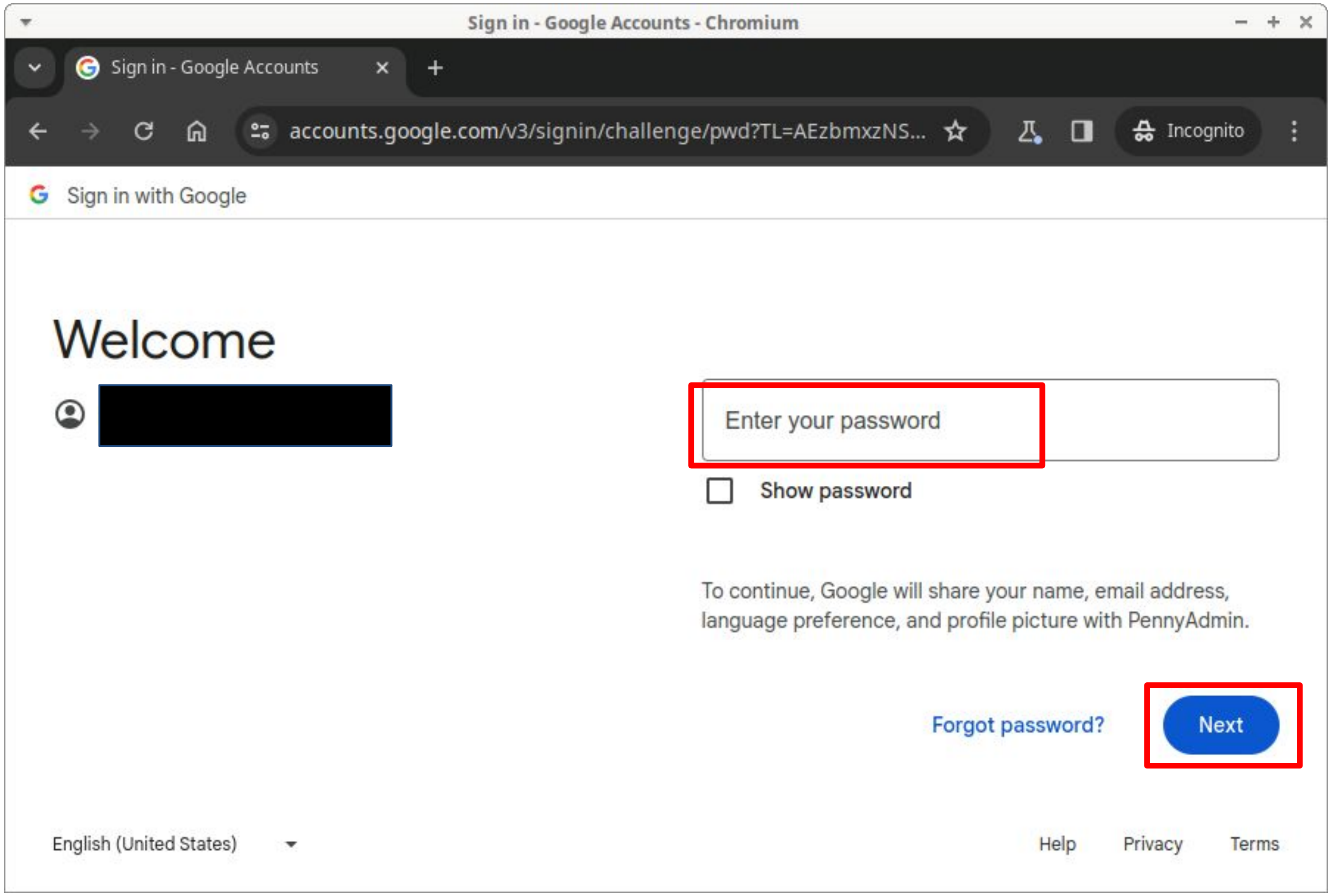
Next

English (United States)

Help Privacy Terms

Google Authentication

- See **PennyAdminGoogle** app (cont.)



Sign in - Google Accounts - Chromium

accounts.google.com/v3/signin/challenge/pwd?TL=AEzbmxzNS...

Sign in with Google

Welcome

[redacted]

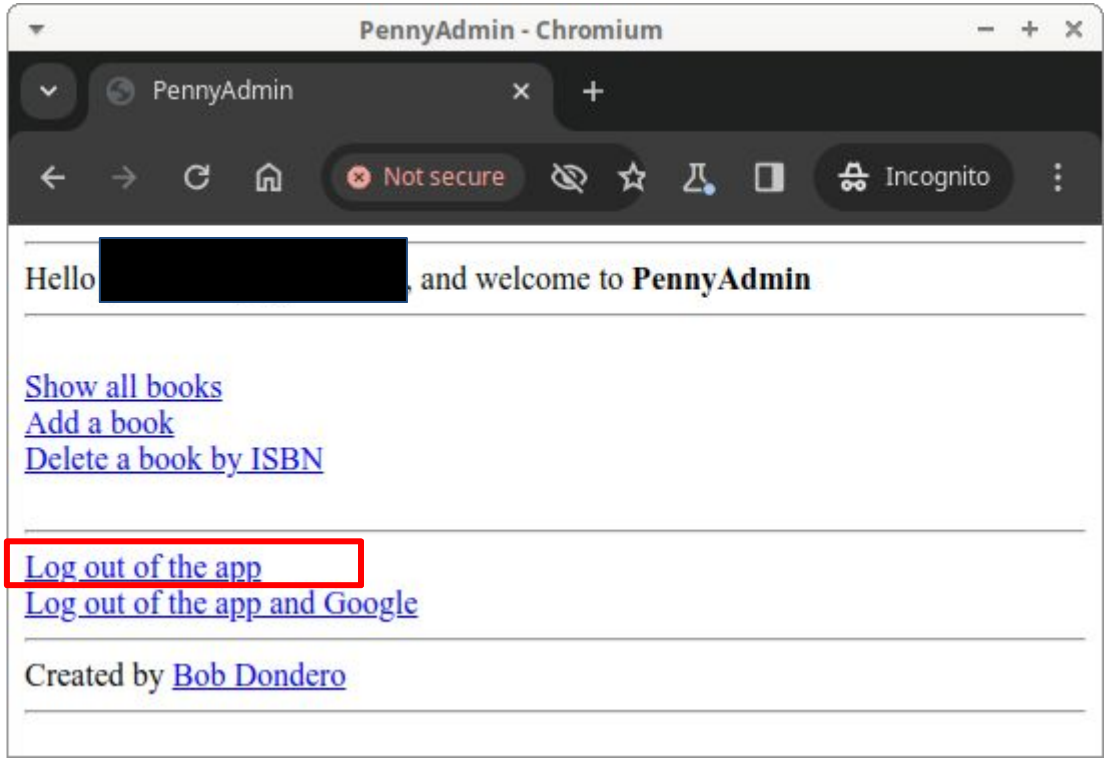
Enter your password

□ Show password

To continue, Google will share your name, email address, language preference, and profile picture with PennyAdmin.

Google Authentication

- See **PennyAdminGoogle** app (cont.)



Hello [redacted], and welcome to **PennyAdmin**

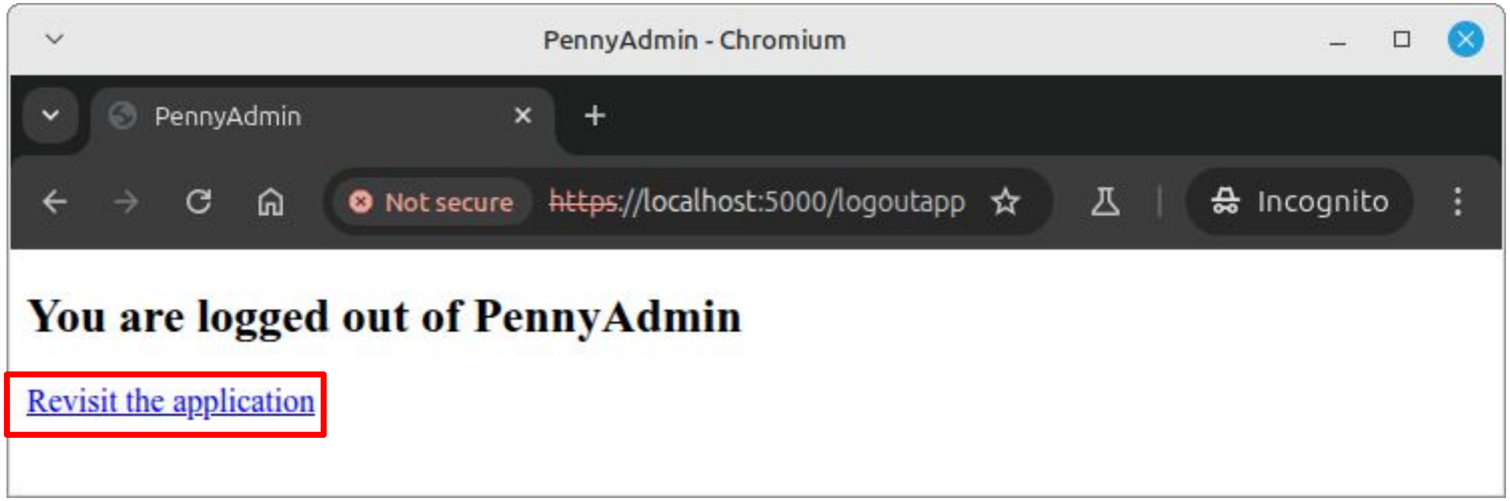
Show all books
Add a book
Delete a book by ISBN

Log out of the app
Log out of the app and Google

Created by Bob Dondero

Google Authentication

- See **PennyAdminGoogle** app (cont.)

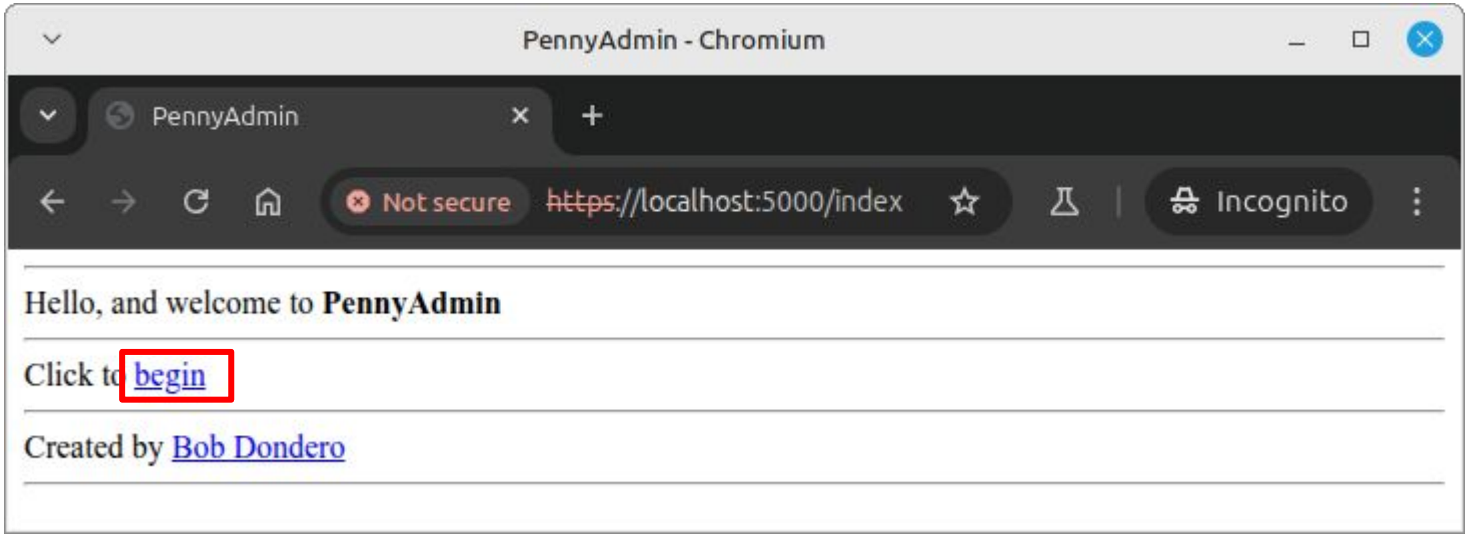


You are logged out of PennyAdmin

Revisit the application

Google Authentication

- See **PennyAdminGoogle** app (cont.)



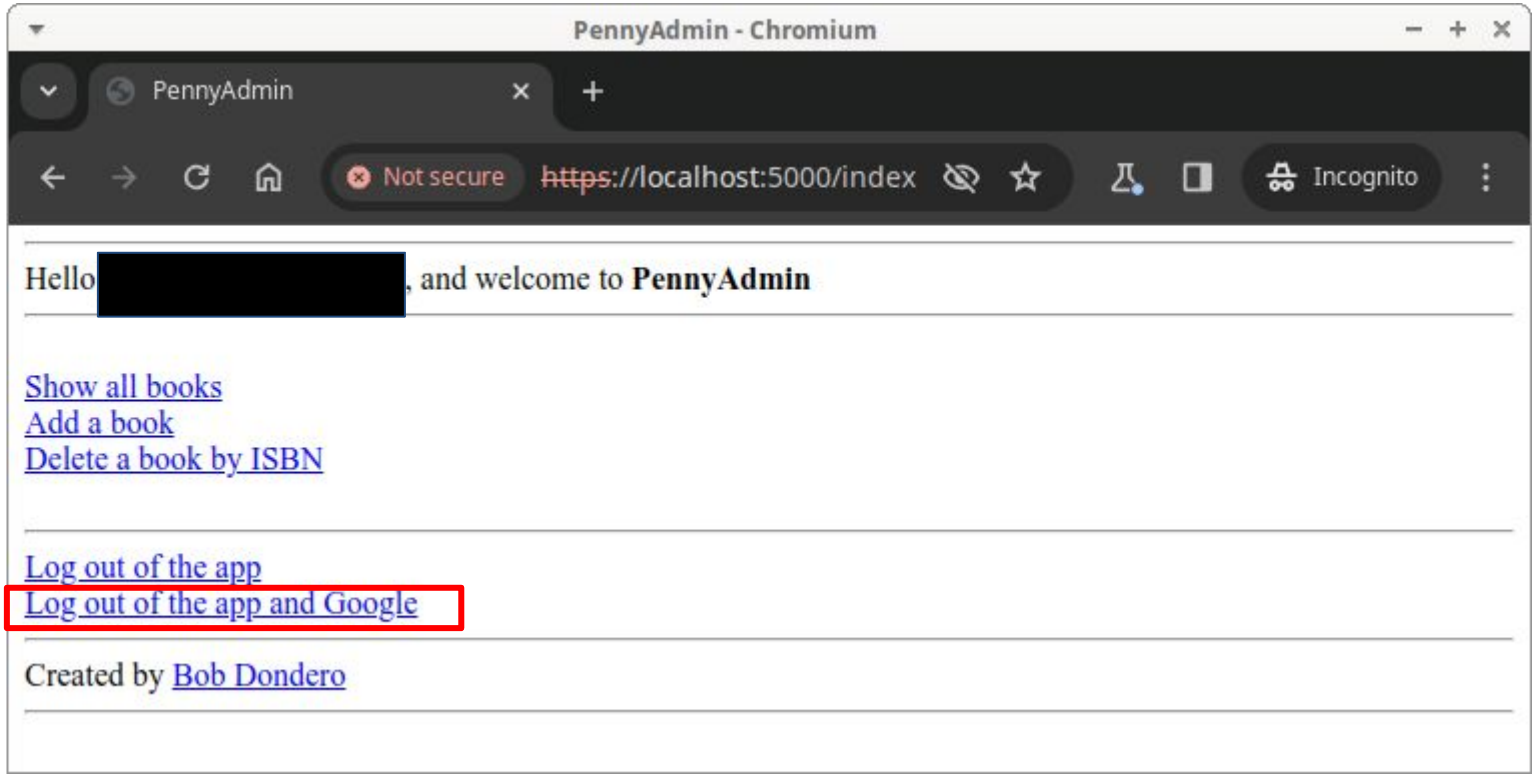
Hello, and welcome to **PennyAdmin**

Click to begin

Created by Bob Dondero

Google Authentication

- See **PennyAdminGoogle** app (cont.)



Hello [redacted], and welcome to **PennyAdmin**

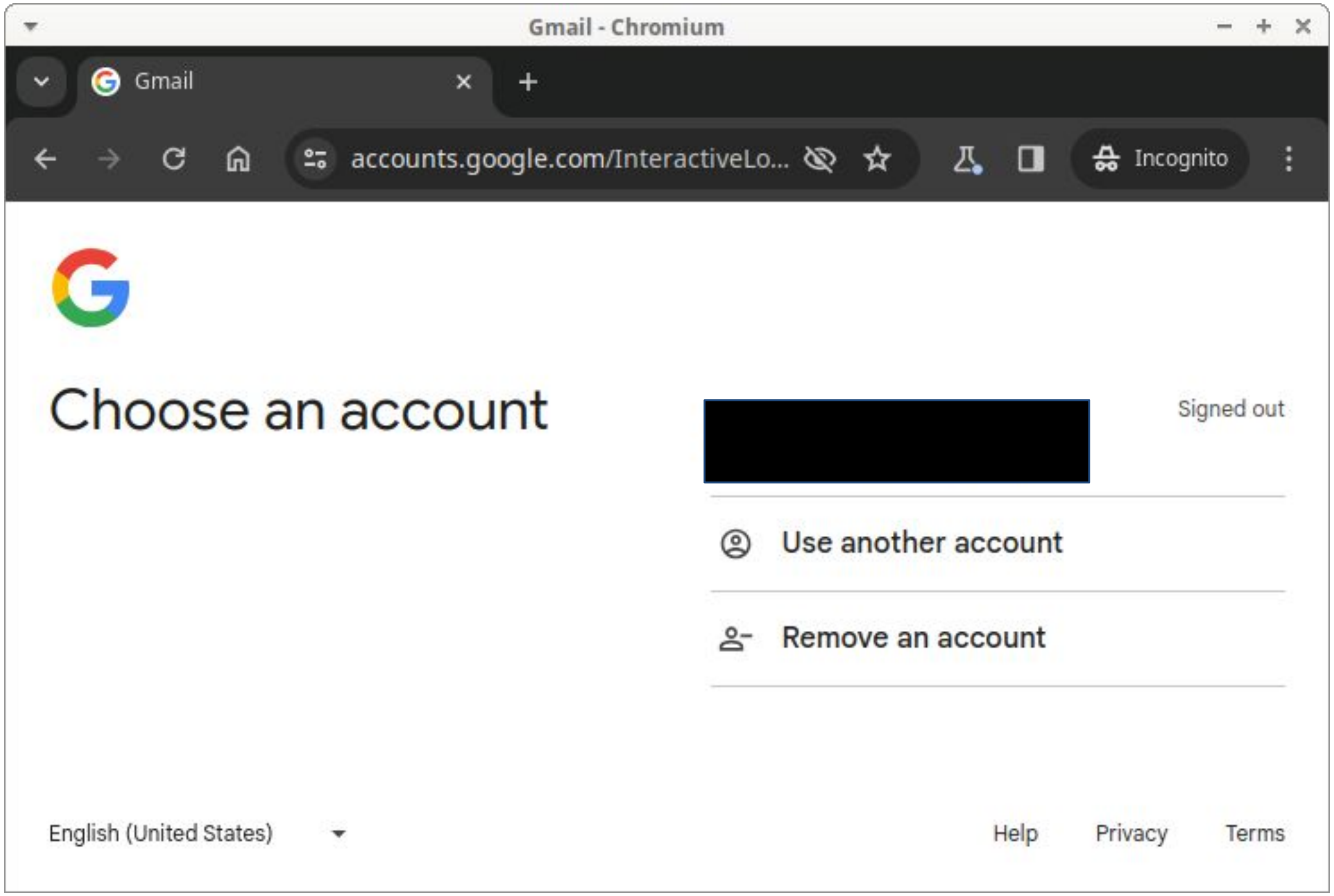
Show all books
Add a book
Delete a book by ISBN

Log out of the app
Log out of the app and Google

Created by Bob Dondero

Google Authentication

- See **PennyAdminGoogle** app (cont.)



Choose an account

[redacted]

Signed out

How to show loggedout page?

Google Authentication

- See **PennyAdminGoogle** app (cont.)
  - See optional lecture material for:
    - The code
    - How to run it on your local computer
    - How to run it on Render
    - How it works

Google Authentication

- **Pros**
  - Users need not remember (yet another) password
  - Application need not manage usernames or passwords
  - Application ***cannot*** access passwords
  - Application can access profile info that user provided to Google
    - Given name, family name, picture, …

Google Authentication

- **Cons**
  - Complex
  - Adds overhead, but mostly only during first user visit per browser session
  - Application is constrained to users who have Google accounts
  - Must use HTTPS with local server
  - If attacker learns user's password for **Google**, then attacker learns user's password for **your app**

Agenda

- Data comm attacks
- Third-party authentication (briefly)
  - CAS
  - Microsoft EntraID authentication
  - Google authentication
  - **Auth0 authentication**

Auth0 Authentication

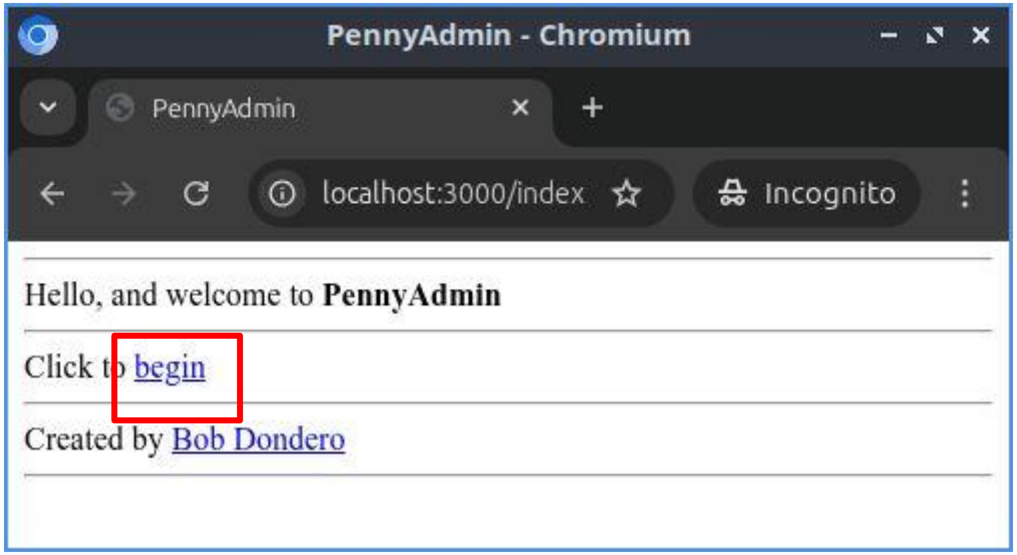
- ***Auth0***
  - https://auth0.com
  - A private company
  - Sells authentication services
  - Has a free tier

Auth0 Authentication

- See **PennyAdminAuth0** app
  - Its behavior…

Auth0 Authentication

- See **PennyAdminAuth0** app (cont.)



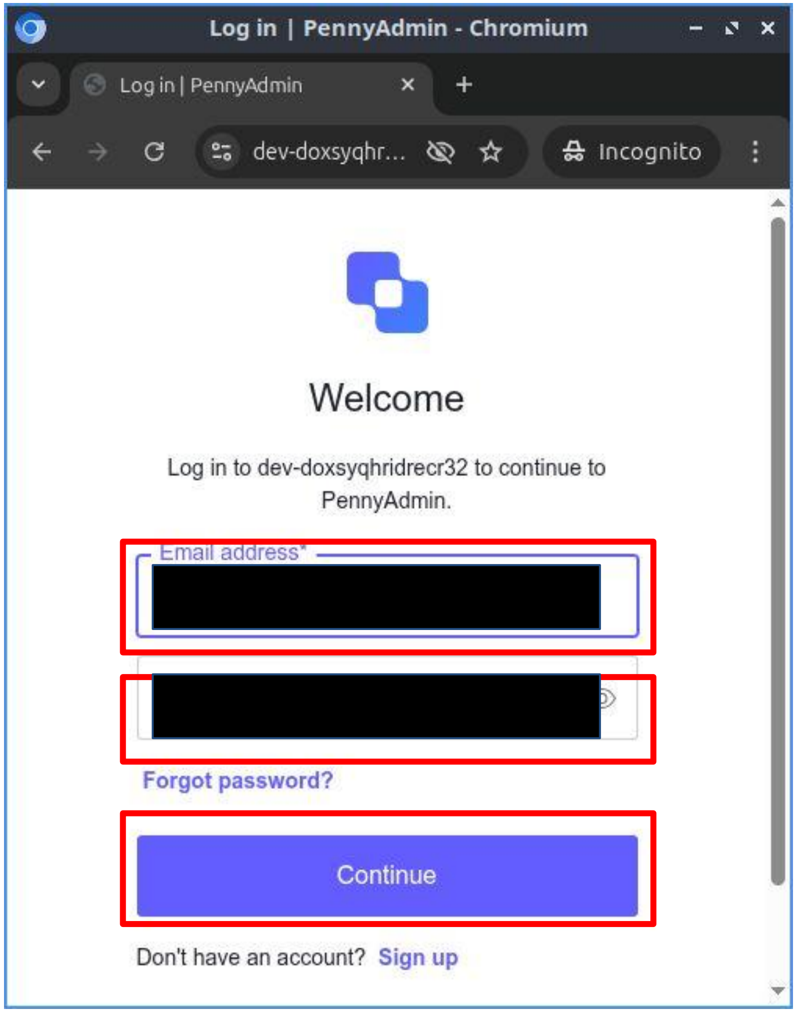
Hello, and welcome to **PennyAdmin**

Click to begin

Created by Bob Dondero

Auth0 Authentication

- See **PennyAdminAuth0** app (cont.)



Welcome

Log in to dev-doxsyqhridrecr32 to continue to PennyAdmin.

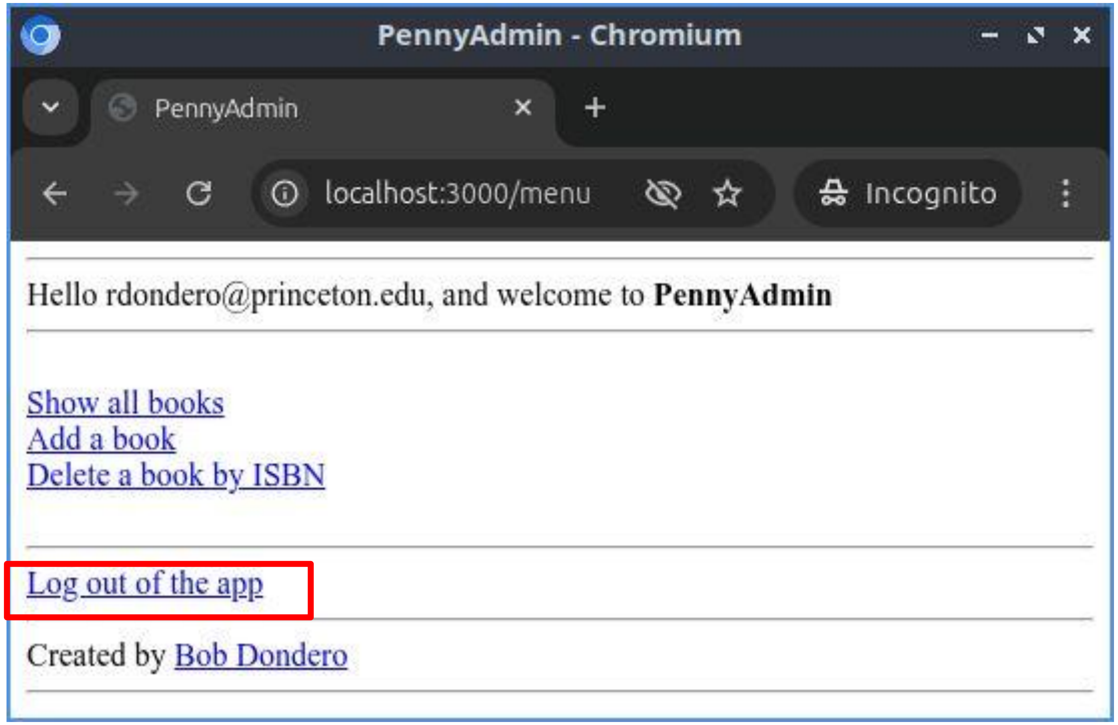
Email address* [redacted]

[redacted]

Don't have an account? Sign up

Auth0 Authentication

- See **PennyAdminAuth0** app (cont.)



Hello rdondero@princeton.edu, and welcome to **PennyAdmin**

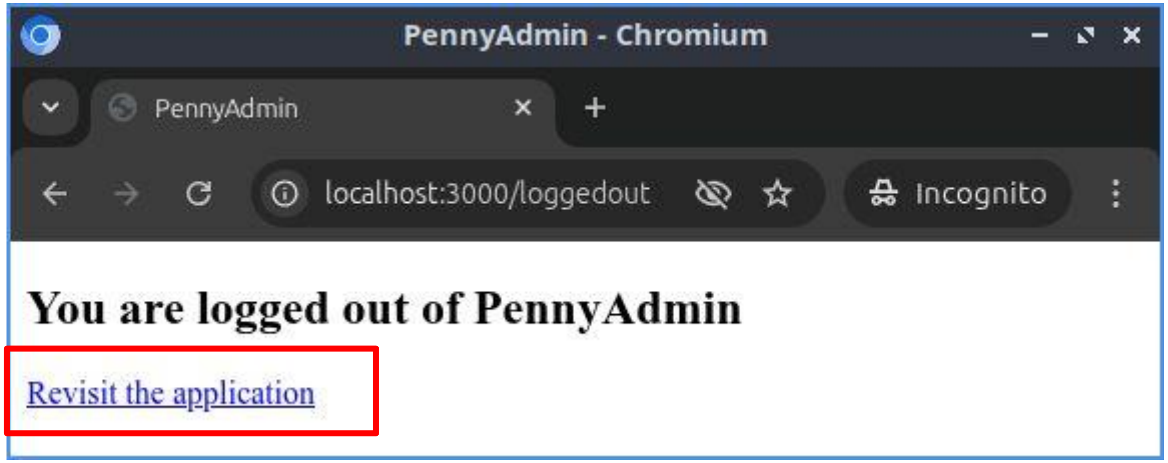
Show all books
Add a book
Delete a book by ISBN

Log out of the app

Created by Bob Dondero

Auth0 Authentication

- See **PennyAdminAuth0** app (cont.)

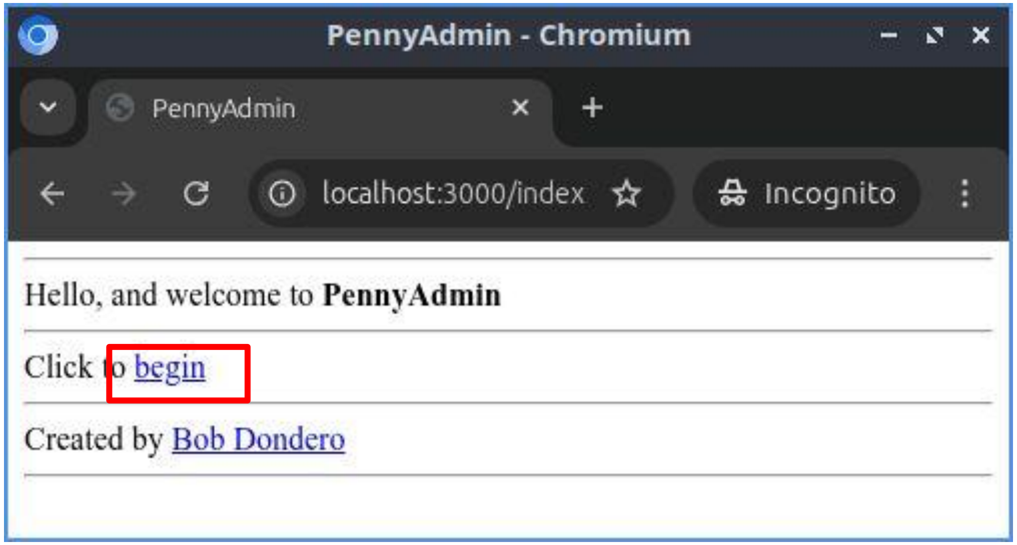


You are logged out of PennyAdmin

Revisit the application

Auth0 Authentication

- See **PennyAdminAuth0** app (cont.)



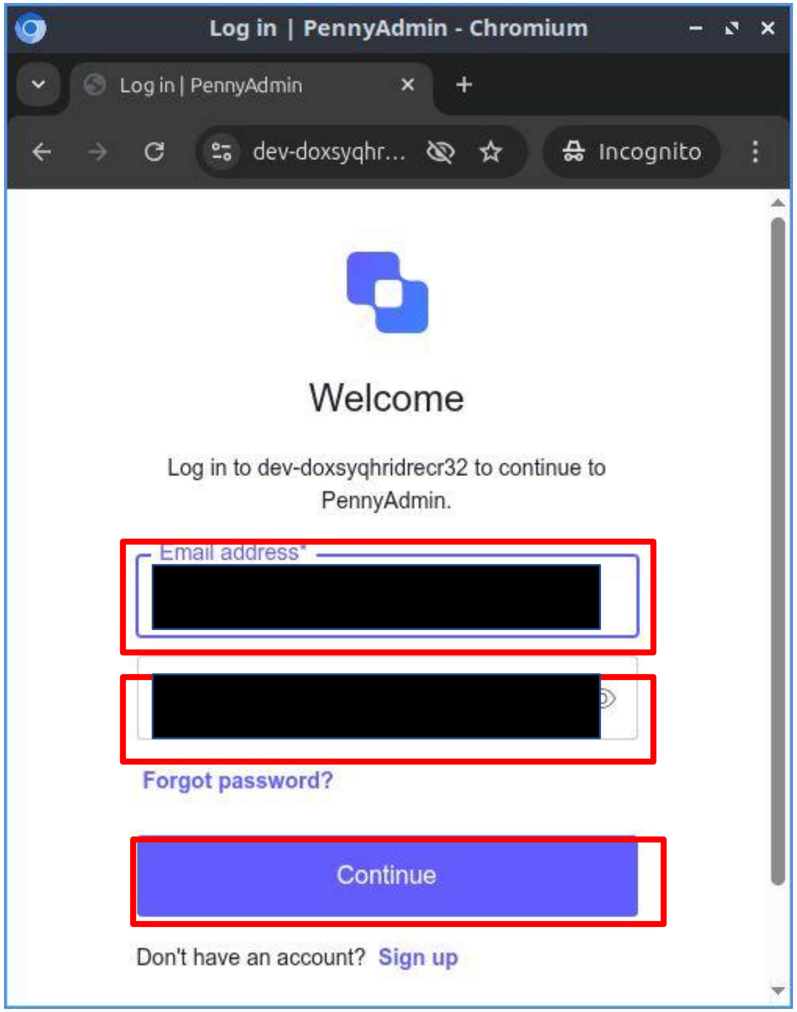
Hello, and welcome to **PennyAdmin**

Click to begin

Created by Bob Dondero

Auth0 Authentication

- See **PennyAdminAuth0** app (cont.)



Welcome

Log in to dev-doxsyqhridrecr32 to continue to PennyAdmin.

Email address* [redacted]

[redacted]

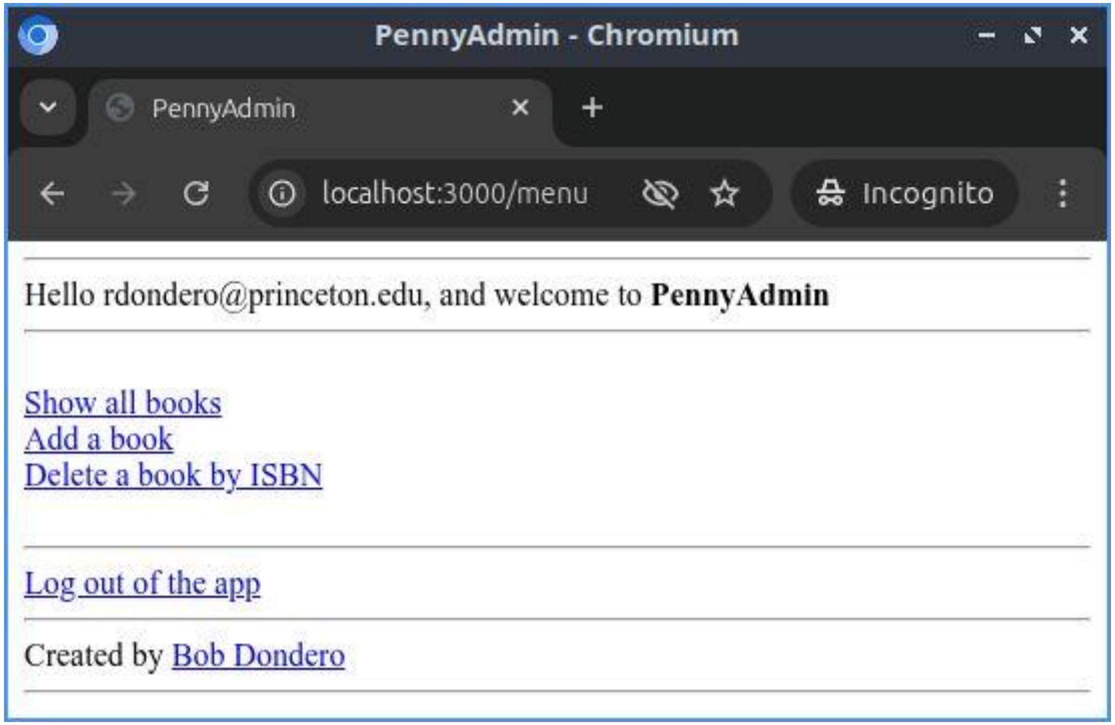
Forgot password?

Continue

Don't have an account? Sign up

Auth0 Authentication

- See **PennyAdminAuth0** app (cont.)



Hello rdondero@princeton.edu, and welcome to **PennyAdmin**

Show all books
Add a book
Delete a book by ISBN

Log out of the app

Created by Bob Dondero

Auth0 Authentication

- See **PennyAdminAuth0** app (cont.)
  - See optional lecture material for:
    - The code
    - How to run it on your local computer
    - How to run it on Render

Auth0 Authentication

- **Pros**
  - Application need not manage usernames or passwords
  - Application ***cannot*** access passwords
  - Application can access profile info that user provided to Auth0
  - Many options available
    - Two-factor authentication
    - Google login alternative
  - Good documentation
  - Users not constrained to Princeton or Google

Auth0 Authentication

- **Cons**
  - Complex
  - Adds overhead, but mostly only during first user visit per browser session
  - Free tier limits login count (but the limit is generous)

Lecture Summary

- In this lecture we covered:
  - Data storage attacks
  - Data comm attacks
  - Third-party authentication (briefly)
    - CAS authentication
    - Microsoft EntraID authentication
    - Google authentication
    - Auth0 authentication

Lecture Series Summary

- In this lecture series we covered:
  - SQL injection attacks
  - Cross-site scripting (XSS) attacks
  - Authentication & authorization
  - Cookie forgery attacks
  - Cookie privacy attacks
  - Cross-site request forgery (CSRF) attacks
  - Data storage attacks
  - Data comm attacks
  - Third-party authentication (briefly)

More Information

- The COS 333 *Lectures* web page provides references to supplementary information