Open standards: Open authentication and Identity Management tool

Decentralised Citizens ENgagement Technologies

Specific Targeted Research Project Collective Awareness Platforms

FP7 - CAPS
Project no. 610349
D-CENT
Decentralised Citizens
ENgagement Technologies

Lead beneficiary: ThoughtWorks

D5.4 Open standards: Open authentication and Identity Management tool / Overview of the OAuth2 Provider and Identity Management tool

July 2015
Version Number: 1

Authors:
Linda Roy (ThoughtWorks)
John Cowie (ThoughtWorks)

Editors and reviewers:
Robert Bjarnason (Citizens Foundation)
Harry Halpin (ERCIM)

The work leading to this publication has received funding from the European Union's Seventh Framework Programme (FP7/2007 – 2013) under grant agreement n° 610349.

The content of this report reflects only the author's view and that the Commission is not responsible for any use that may be made of the information it contains.

Contents

1. Executive summary

The focus of D5.4 is to create a tool that can be easily deployed by democratic organisations, to provide a secure single sign-on (SSO) service for their users. This SSO service can be easily integrated with other tools hosted by these organisations (such as the policy-drafting tool of D5.3) to provide access to the tools.

Furthermore the use of a consistent protocol implementation (OAuth2) allows organisations to share their userbases. To make a tool available to the users of another organisation, the tool can simply be registered as a client application to the OAuth2 instance belonging to that organisation.

In-progress work to enable the administration of user permissions will allow organisations to ensure that their tools are only available to trusted members.

The development name of the OAuth2 tool is Stonecutter, which will be used to refer to the application in the remainder of this document.

During development an up-to-date deployed instance of Stonecutter can be found at
https://sso.dcentproject.eu

The code is open sourced and currently available at
https://github.com/ThoughtWorksInc/Stonecutter

High Level Feature Overview

Completed Stonecutter features:

- User registration
- User sign in
- User sign in from a client application
- Client application permissions
- Manage app permissions in profile page
- Change password
- Delete Account
- Integration with Policy-drafting tool

2. User Registrtion

Feature definition:

Users can register an account with Stonecutter. User credentials are securely stored in the database.

User Story:

As a user of a tool in the D-CENT platform

I want to register on Stonecutter

So that I have a Stonecutter account that I can use to access other tools

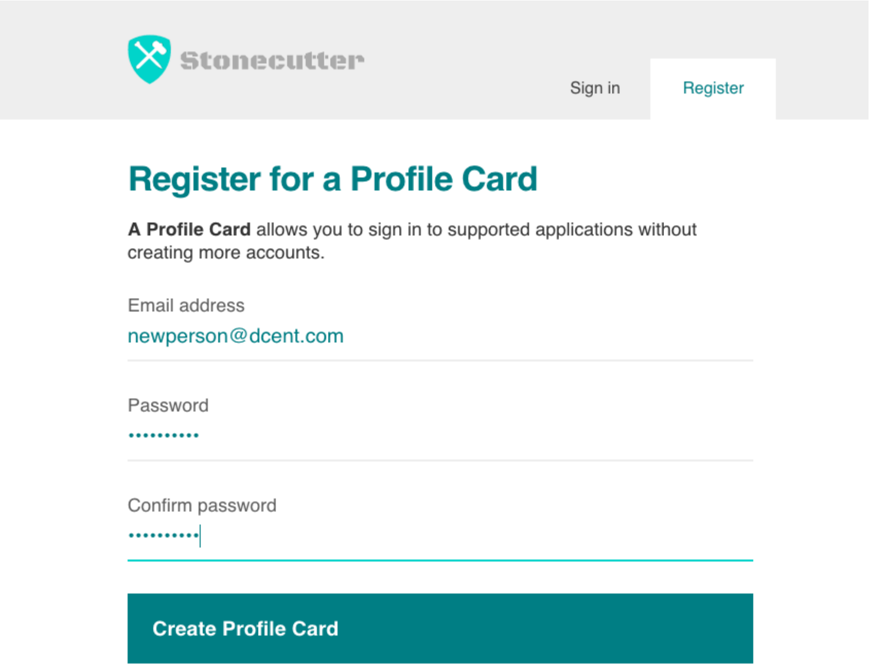
Description:

Users of the tools in D-CENT platform can register an account with Stonecutter using their email address and a password. A 'Profile Card' metaphor is used to refer to the user's account and identity.

Technical implementation:

When the form is submitted, user details are validated on the server and stored in the database (MongoDB). The user's password is salted and hashed with bcrypt before being stored in the database.

Screenshots:



Register for a Profile Card

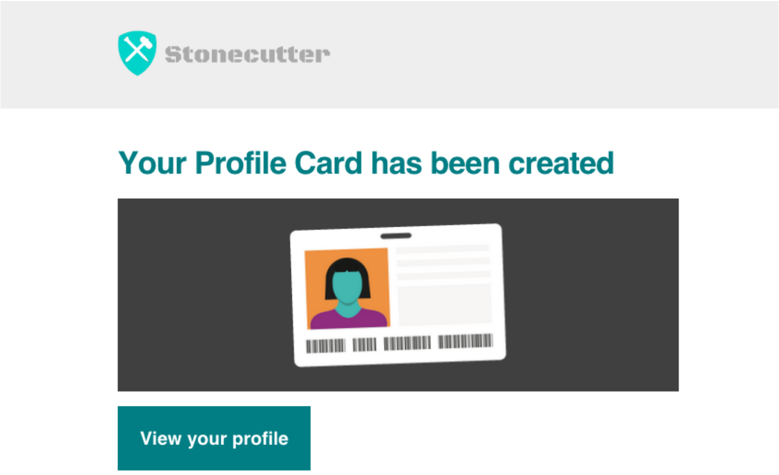
A Profile Card allows you to sign in to supported applications without creating more accounts.

Email address
newperson@dcent.com

Password
••••••••••

Confirm password
••••••••••

Figure 1: Registration screenshot 1



Your Profile Card has been created

Figure 2: Registration screenshot 2

3. User Sign-in

Feature definition:

Users can sign in to Stonecutter to view their profile and modify any user details.

User Story:

As a registered user of Stonecutter

I want to sign in

So that I can view my profile

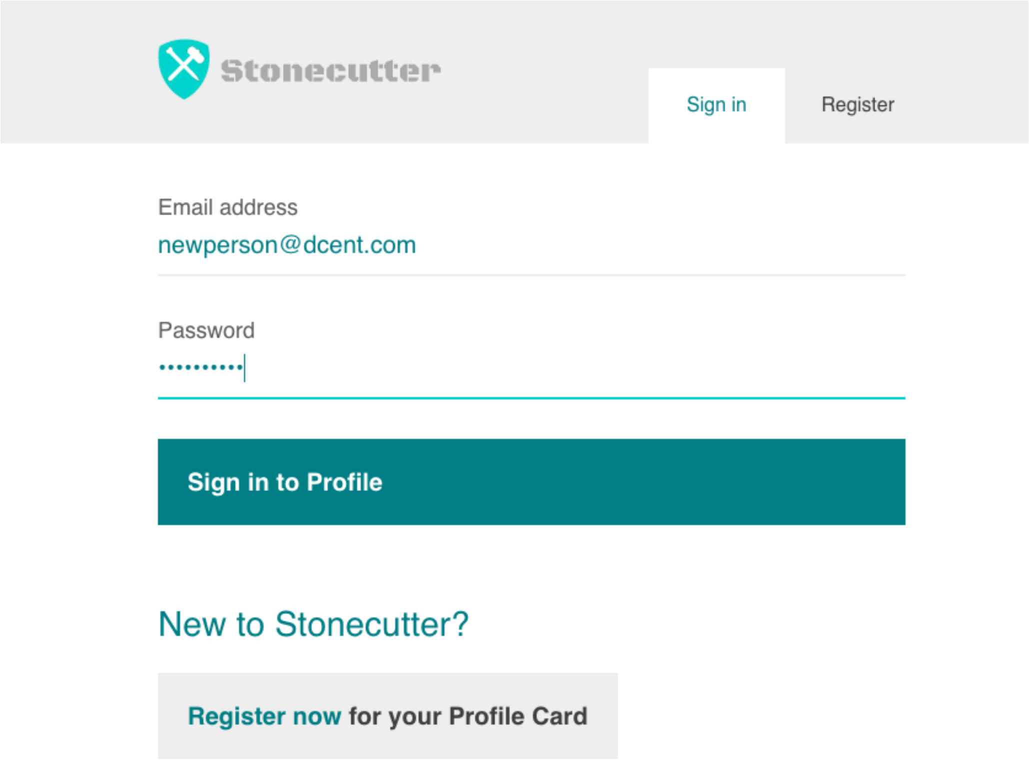
Description:

Registered users can sign in to Stonecutter with their email address and password. If either the email address or password is incorrect an error message will be shown. They can sign out of their account when they choose.

Technical implementation:

Clicking 'Sign in to Profile' creates a form post to the server. The username and password are validated using the Clauth OAuth2 library. If the sign-in attempt is successfully validated then the user's ID is added to a session, backed by a browser-cookie. In subsequent requests the server will check the session to identify whether a valid user is signed in. When the sign-out button is clicked, the User's ID is removed from the session. Subsequent page requests to user-specific pages will redirect the user to the sign-in page when a User ID is not present in the session.

Screenshots:



Email address
newperson@dcent.com

Password
••••••••••

Figure 3: User sign-in screenshot 1

4. Sign-in to client application

Feature definition:

When a user clicks on 'sign-in' on an integrated client application, the client can log the user in using identity attributes returned by Stonecutter.

User Story:

As a user of a tool belonging to my organisation

I want to be able login with my Stonecutter account

So that I am able to access the tool without a mainstream social-media account.

Description:

When the user clicks the 'sign-in' button on a client application, the client redirects the user to Stonecutter. If the user is not signed in, Stonecutter will request that the user either signs in, or registers a new account. Stonecutter will then interact securely with the client to ensure that the client's request is valid, and in response will return the signed-in user's ID and email address to the client.

Technical Implementation:

The interaction between Stonecutter and the client adheres to the OAuth2 protocol (described at https://tools.ietf.org/html/rfc6749). Clauth, a Clojure OAuth2 library, is used to set up the OAuth2 endpoints for handling authorisation and token requests.

5. Client application permissions

Feature definition:

When users sign in or register from an integrated client app, they are presented with a choice to either allow or deny the application permission to access their account.

User Story:

As a user signing in/registering to Stonecutter via a client app

I want to have the option to allow or deny the app to get my data

So that I am able to choose which applications I share my profile with

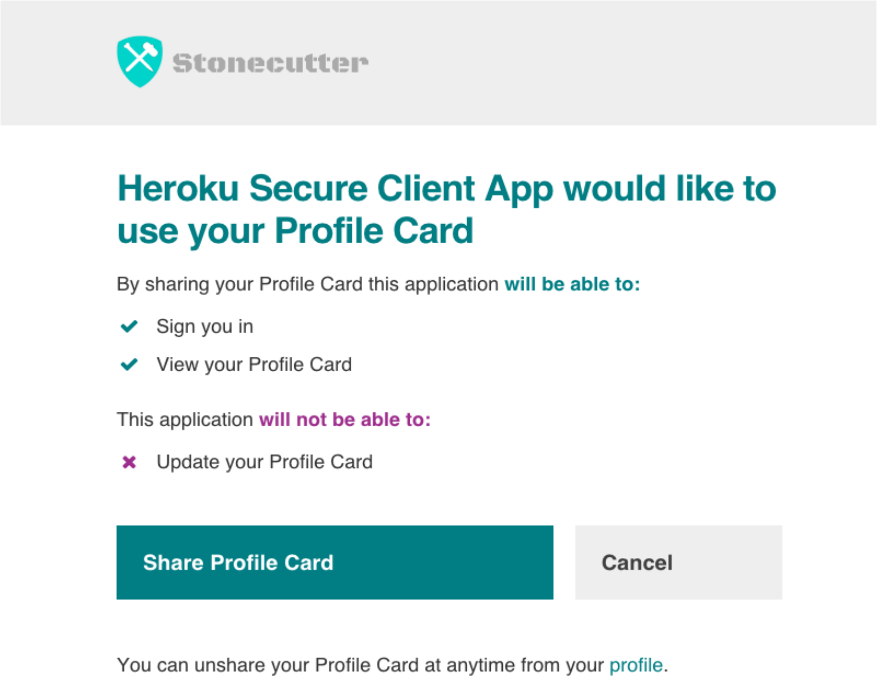
Description:

When the user is redirected to Stonecutter from a client application, once they have signed in they are shown a screen that allows them to grant the client application access to their profile. The terminology used is 'Share Profile Card' for allowing application access, and 'Cancel' for denying application access. If the user chooses 'Share Profile Card', then they will be redirected to the client app to continue the action they wanted to do. If the user chooses to deny permission to the client application then the user will be shown a screen confirming this decision. When they click to return to the client they will be redirected back to the client with an error notification in the redirect, so that the client is unable to continue the sign-in process.

Technical implementation:

If a user grants permission to an application, the client ID of this application is stored in a permissions list in that user's record in the database. In subsequent sign-ins, the database is checked for the presence of the client ID in the user's permissions list, and if it is found the Allow/Deny screen is bypassed.

Screenshots:



Heroku Secure Client App would like to use your Profile Card

By sharing your Profile Card this application **will be able to:**

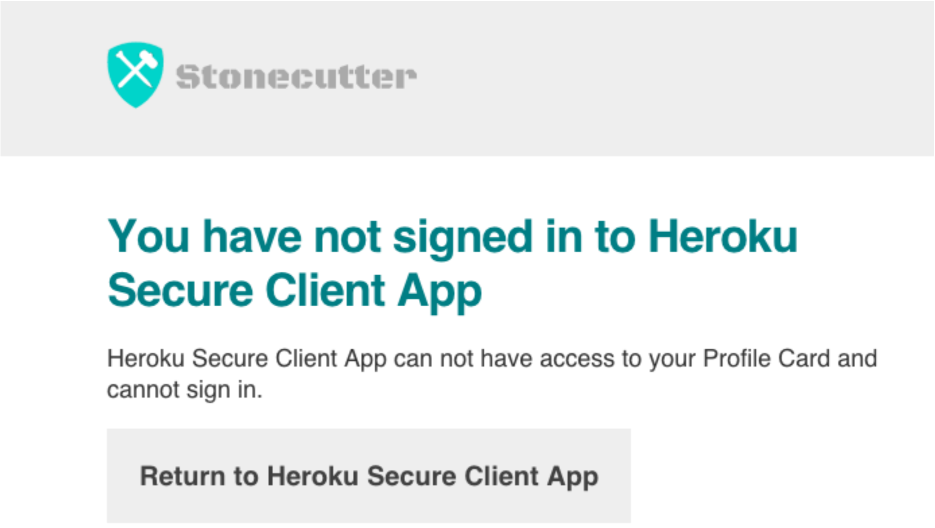
- ✓ Sign you in
- ✓ View your Profile Card

This application **will not be able to:**

- ✗ Update your Profile Card

You can unshare your Profile Card at anytime from your profile.

Figure 4: Client application permissions screenshot 1



You have not signed in to Heroku Secure Client App

Heroku Secure Client App can not have access to your Profile Card and cannot sign in.

Figure 5: Client application permissions screenshot 2

6. Manage app permissions in profile page

Feature definition:

Users can manage their application permissions from their profile page.

User Story:

As a user of Stonecutter

I want to be able to revoke application permissions

So that I have control over which applications have access to my profile

Description:

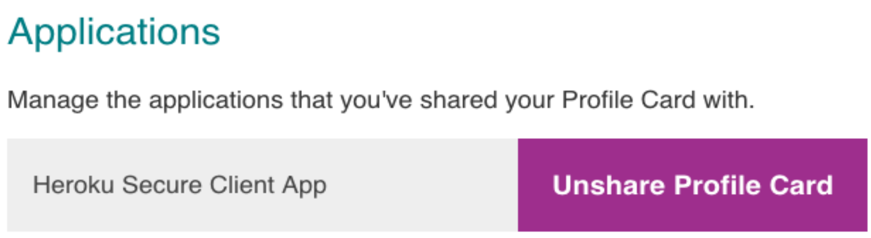
From the user profile page on Stonecutter, the user is able to view a list of applications that they have granted access to their profile. They are also able to revoke a client application's access to their profile. When the user clicks 'Unshare Profile Card', the user is taken to a page to confirm this action. If the user decides to confirm 'unshare profile card' they are taken back to their profile page and the subscribed app will not be seen in the list of applications subscribed to. If the user clicks on 'Cancel' then they are taken back to profile page and the subscribed app can be seen with the list of other applications subscribed to.

If user decides they want to sign in to an application that has had access revoked, they will have to grant the application permission once again.

Technical implementation:

When the 'Unshare Profile Card' button is clicked on the confirmation page, a form post triggers the removal of the corresponding client ID from the permission list in the user record in the database.

Screenshots:

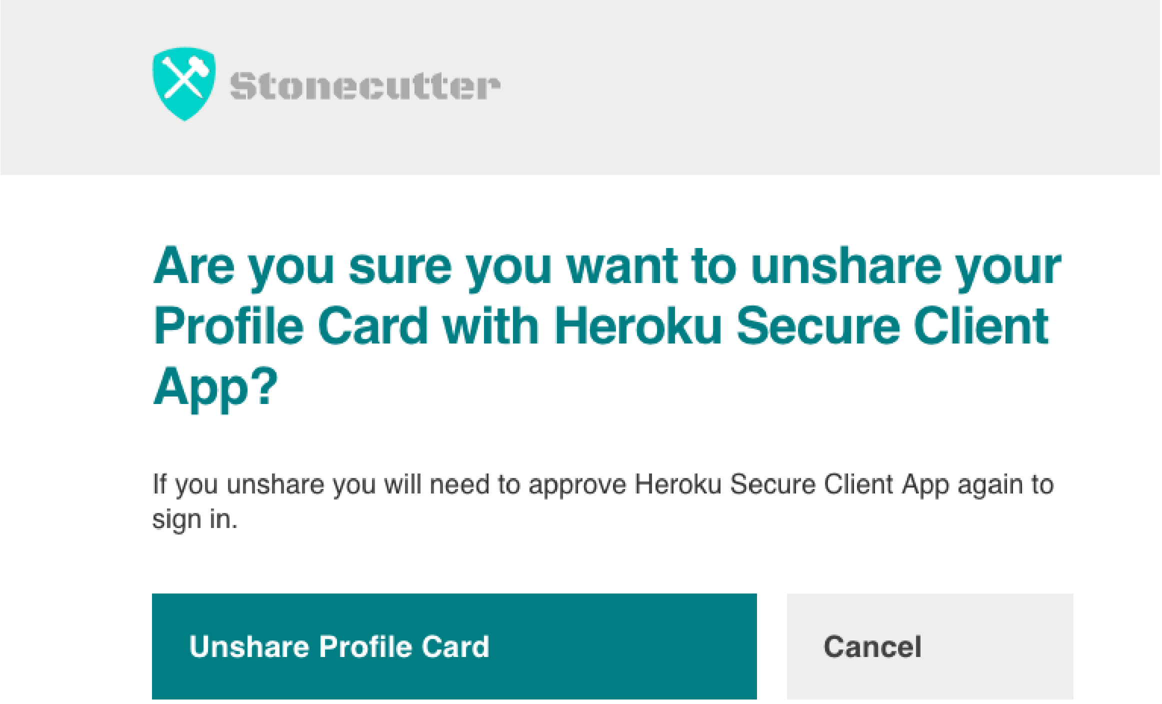


Applications

Manage the applications that you've shared your Profile Card with.

Heroku Secure Client App

Figure 6: Manage app permissions in profile page screenshot 1



Are you sure you want to unshare your Profile Card with Heroku Secure Client App?

If you unshare you will need to approve Heroku Secure Client App again to sign in.

Figure 7: Manage app permissions in profile page screenshot 2

7. Change Password

Feature definition:

Users can change their password from their profile page

User Story:

As a user of Stonecutter

I want to be able to change my password

So that I am able to keep my account secure

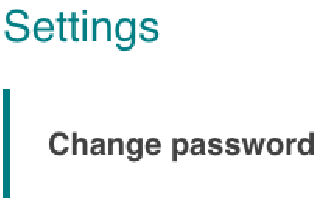
Description:

Users have an option to change their password from their Stonecutter profile page by clicking 'Change password' in the 'Settings' section. The user enters their current password and new password and clicks save. The next time the user wants to sign in, they have to use the new password.

Technical implementation:

Clicking 'Save changes' will trigger a web-form post to the server. The user's existing password and new password will be validated server side.

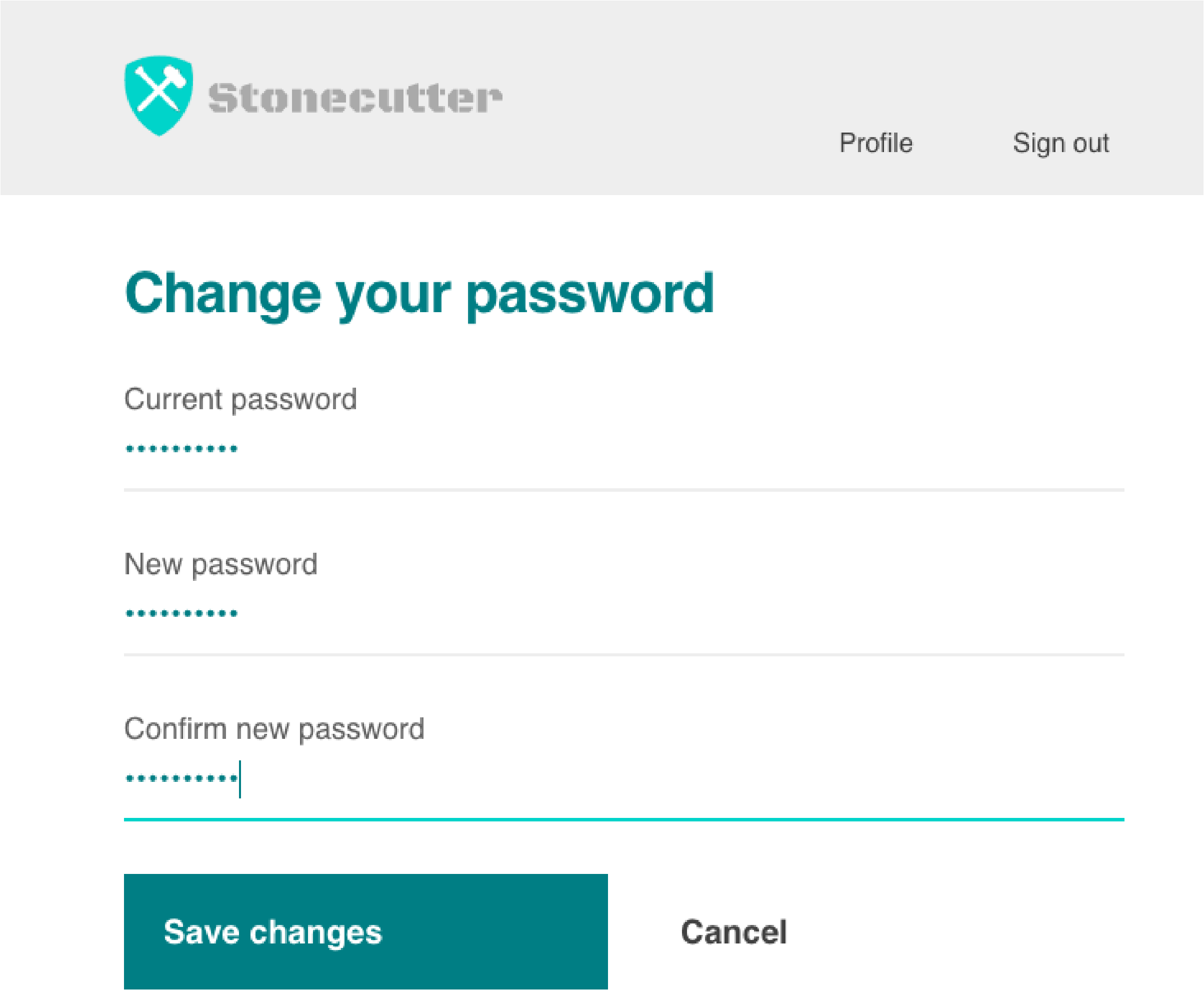
ScreenShots:



Settings

Change password

Figure 8: Change password screenshot 1



Change your password

Current password

••••••••••

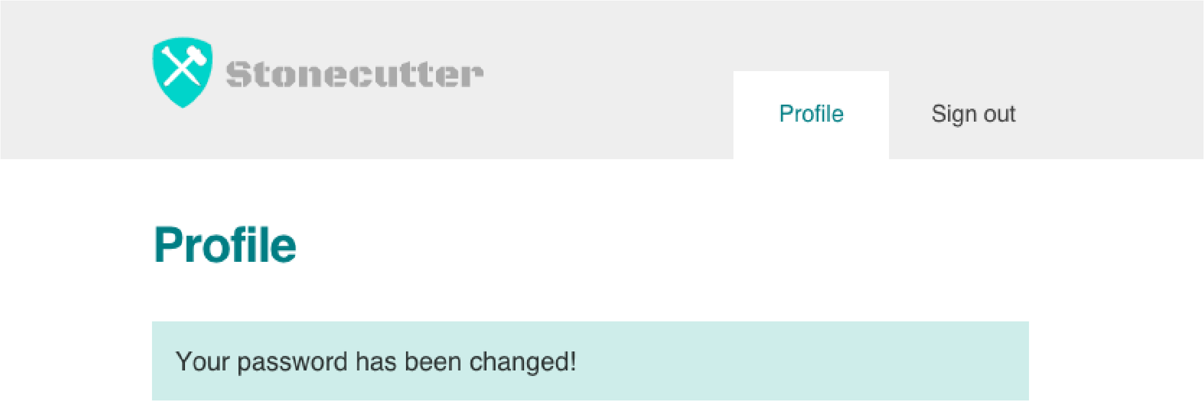
New password

••••••••••

Confirm new password

••••••••••

Figure 9: Change password screenshot 2



Profile

Your password has been changed!

Figure 10: Change password screenshot 3

8. Delete Account

Feature definition:

Users are able to remove their account from Stonecutter.

User Story:

As a user of Stonecutter

I want to be able to delete my account

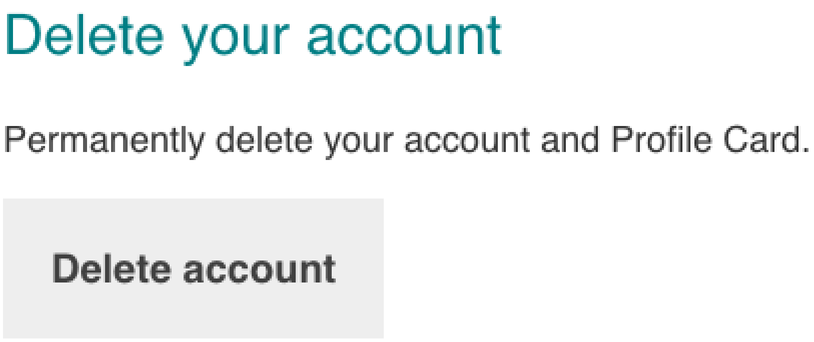
So that Stonecutter no longer has possession of my data

Description:

Users have an option to delete their account from the profile page in Stonecutter. Deleting their account will remove their user data. If the user wishes to sign in via Stonecutter in future, they will have to register a new account.

Technical implementation: If the user chooses to delete their account, their user record is removed from the database and they are logged out of Stonecutter (i.e their existing session is also is removed).

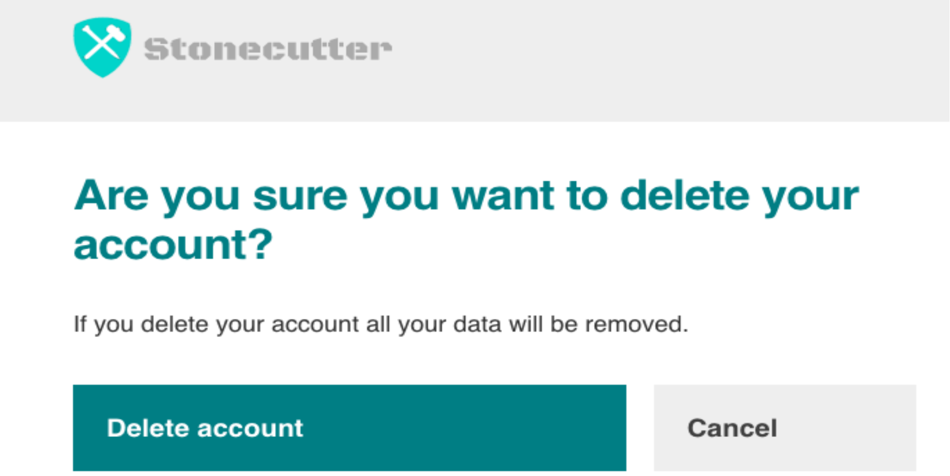
Screenshots:



Delete your account

Permanently delete your account and Profile Card.

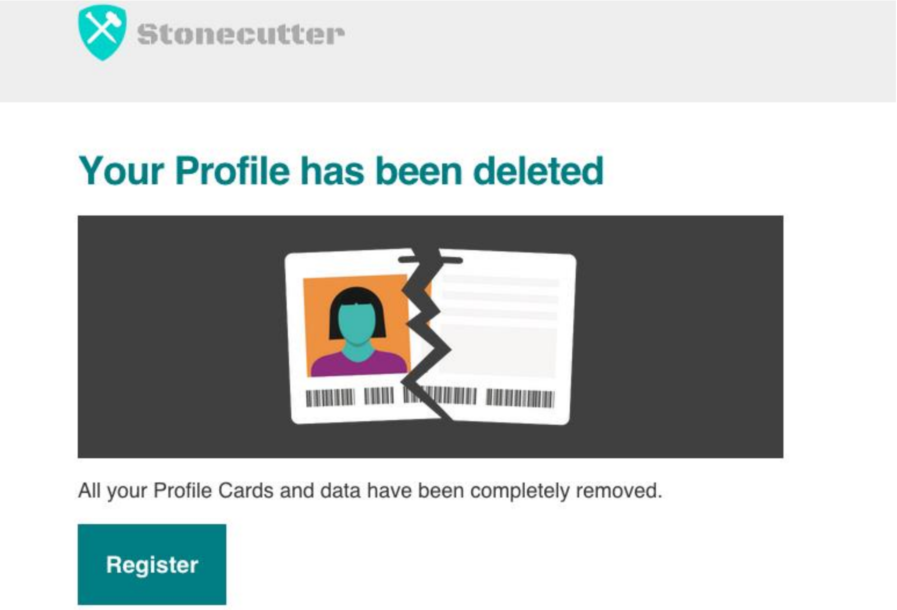
Figure 11: Delete account screenshot 1



Are you sure you want to delete your account?

If you delete your account all your data will be removed.

Figure 12: Delete account screenshot 2



Your Profile has been deleted

All your Profile Cards and data have been completely removed.

Figure 13: Delete account screenshot 3

9. Integration with Policy-drafting tool

Feature definition:

Stonecutter is integrated with Objective8 for the purposes of demonstrating the use of Stonecutter as the basis of a platform of D-CENT tools.

User Story:

As a user of Objective8

I want to be able to sign in with my Stonecutter account

So that I no longer require a Twitter account to use Objective8

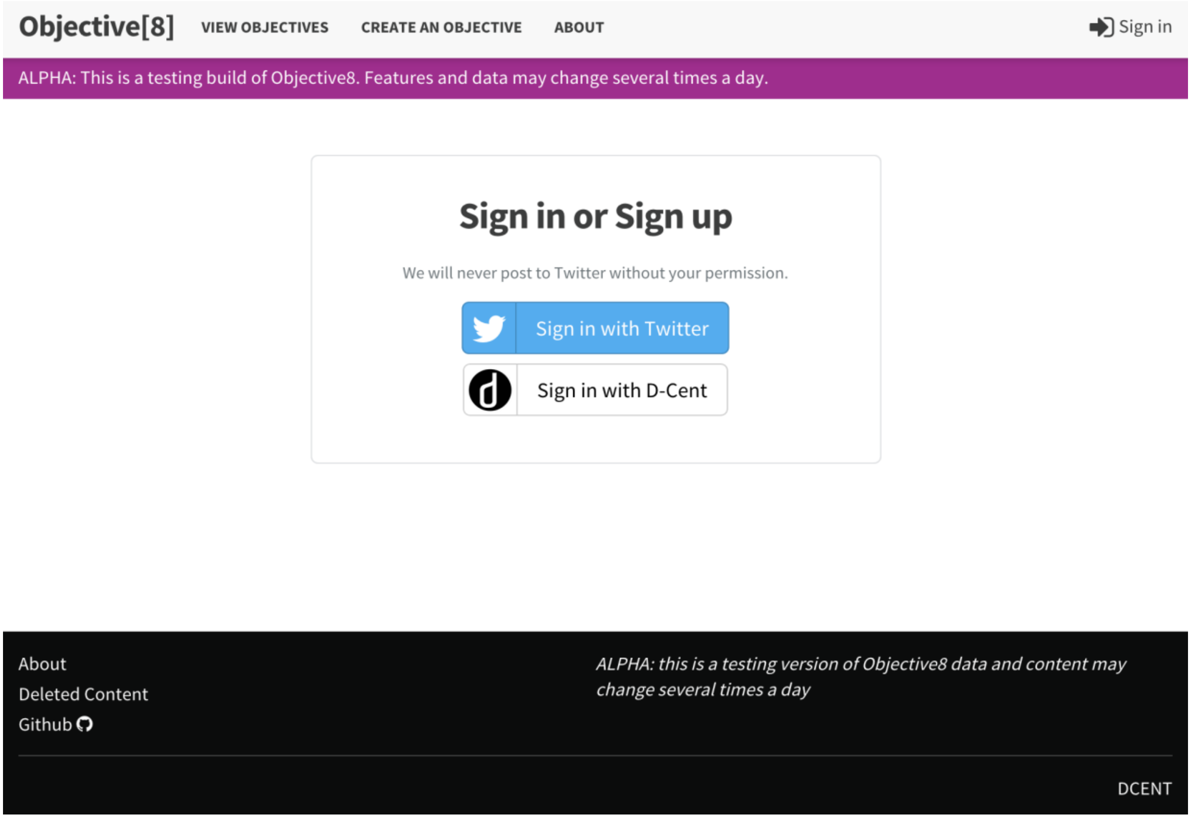
Description:

When visiting Objective8, users are now presented with the option of signing in with a D-CENT-branded version of Stonecutter, alongside Twitter. When choosing this option, the user can successfully access Objective8 with their Stonecutter account.

Technical implementation:

Objective8 was modified to support OAuth2, and configured with the URL of the Stonecutter instance. A Client ID and Secret was shared between the two applications to register Objective8 with the Stonecutter instance.

Screenshots:



ALPHA: This is a testing build of Objective8. Features and data may change several times a day.

Sign in or Sign up

We will never post to Twitter without your permission.

ALPHA: this is a testing version of Objective8 data and content may change several times a day

Figure 14: Integration with Policy-drafting tool screenshot 1

10. User Journey

1) A user first arrives at Stonecutter:
   a) If the user is already logged in, then they are moved to the next step.
   b) If the user does not have an account then they register a new account and progress.
   c) If the user has an account then they log in with their username and password and progress.
2) Once the user is logged in:
   a) If they have visited the Stonecutter site directly they are taken to their profile page.
   b) If they have been redirected from a client application:
      i) If the user has previously granted access to the application then they are redirected back to the client.
      ii) If the user hasn't previously granted access to the application they choose to Allow or Deny access to the client application and redirected back to the client.

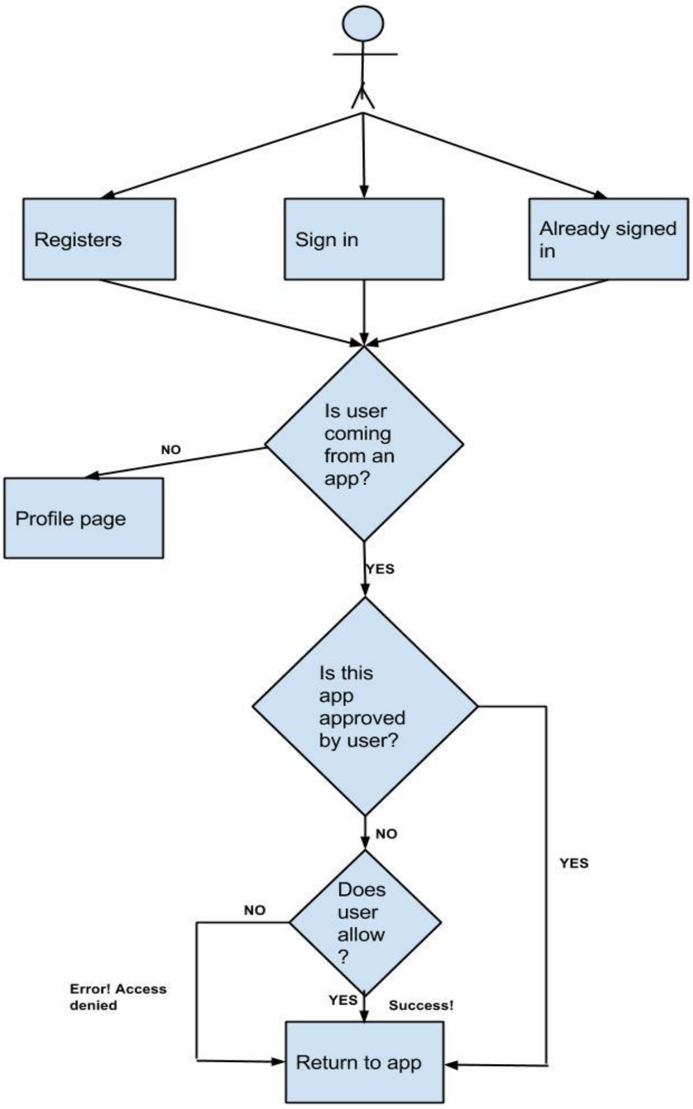


Figure 15: User journey diagram

11. User Testing

User testing was done in the early stages of developing Stonecutter. A series of usability testing was conducted to find out how the users react to a different look and feel of Stonecutter when signing in via a client app.

Tests were conducted with a demo Green Party application.

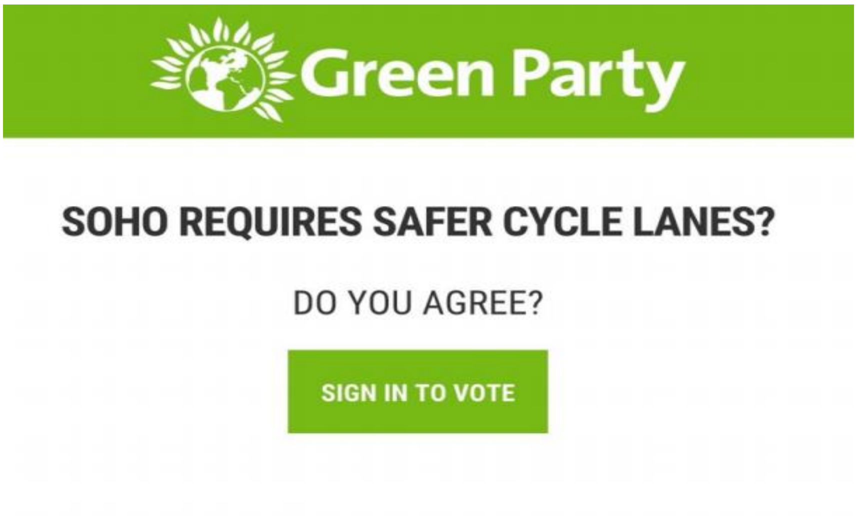


Figure 16: Green part application screenshot

The Green Party application offers participants the opportunity to vote. In order to vote, participants must sign in or register an account with Stonecutter. In the first test instance the user was redirected from the client (Green Party) application to a Stonecutter site with the default Stonecutter branding.

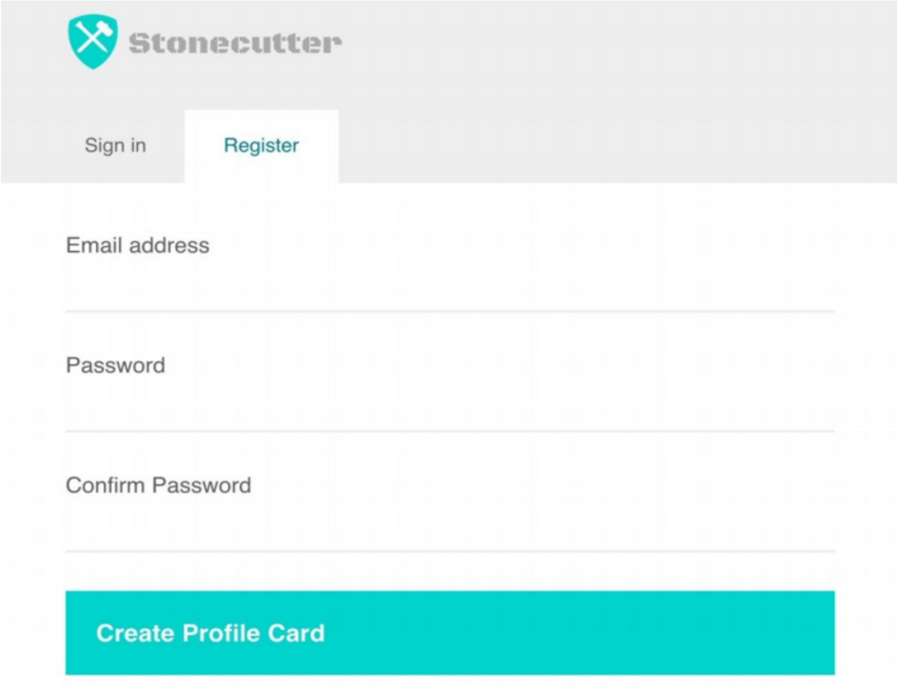


Figure 17: Stonecutter branding registration screenshot

The second test instance the default Stonecutter branding was replaced with the Green Party logo and look and feel, i.e. a branding that matched that of the client application.

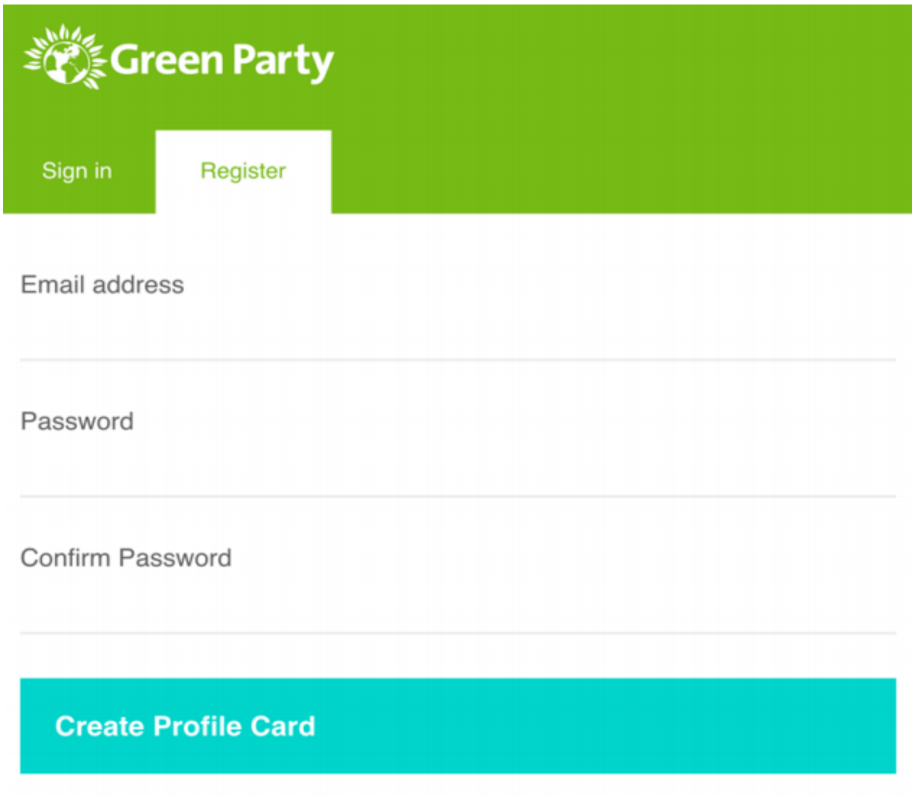


Figure 18: Green party branding registration screenshot 1

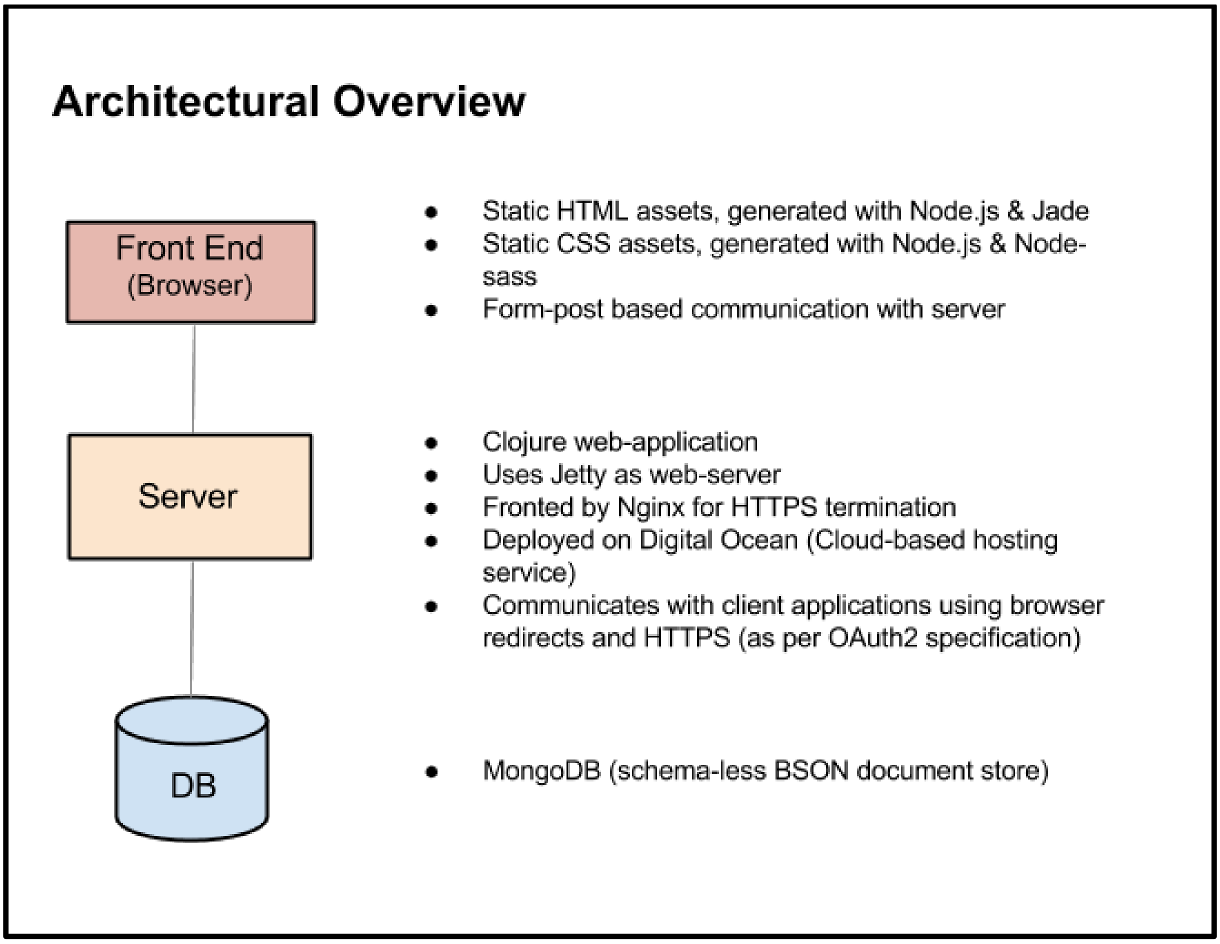
The outcome of these branding tests confirmed that users had a much higher level of trust in a SSO service that was branded to match the organisation that owned the client application. Consistency of branding between different tools which users could access with their account reduced confusion about who held their personal data. As a result of this testing, some work to develop an easy-mechanism for white-labeling Stonecutter with organisational branding was added to the backlog and is currently in development.

11.1 Future User Testing

Users don't derive value from the Stonecutter tool in isolation, but will benefit from using it when they are able to access other tools in the D-CENT platform using a Stonecutter account. As a result more detailed user testing will be performed in conjunction with user testing of the Policy-Drafting (D5.1) and Secure Notifications (D5.6) tools.
Stonecutter is already integrated with the D-CENT-hosted instance of Objective8 (https://objective8.dcentproject.eu), so users are able to try out accessing Objective8 with a Stonecutter account.

The development of the Secure Notifications tool will also include integration with Stonecutter. User testing of the pilot in Finland will include testing the user flow of signing into the tool with a Stonecutter account.

12. Architectural Design



Architectural Overview

Front End
(Browser)

- Static HTML assets, generated with Node.js & Jade
- Static CSS assets, generated with Node.js & Node-sass
- Form-post based communication with server

Server

- Clojure web-application
- Uses Jetty as web-server
- Fronted by Nginx for HTTPS termination
- Deployed on Digital Ocean (Cloud-based hosting service)
- Communicates with client applications using browser redirects and HTTPS (as per OAuth2 specification)

DB

- MongoDB (schema-less BSON document store)

Figure 19: Architectural Design

13. Technology Rationale

Clojure:

- JVM-based language, allowing for familiar deployment
- Expressive dynamically-typed language with an emphasis on high-quality code principles such as immutability by default
- Strong and growing open-source community
- Easy Java interoperation, enabling use of large body of existing Java libraries
- Good level of experience with the ThoughtWorks team, allowing the team to deliver features quickly and at a high-level of quality

MongoDB:

- High level of adoption in industry, so well supported
- Has a shallow initial learning curve
- Easy to deploy
- Schema-less JSON document store making it flexible for user records with potentially different subsets of optional profile data
- Designed for scalability and resiliency

Sass:

- More powerful syntax for producing CSS
- Allows removal of duplication in CSS with variables

14. Security

The development team participated in researching potential vulnerabilities when implementing OAuth2 to understand and mitigate potential vulnerabilities.

In addition the pilot instance of the application is deployed behind HTTPS so that all OAuth2 requests and responses are encrypted. This is a requirement of the OAuth2 specification and removes a set of vulnerabilities.

To ensure the security of the Stonecutter application, the OWasp Top Ten guide was reviewed and followed.

16. Deployment

Ansible is used to provision virtual machines with the dependencies that the Stonecutter application requires, including the configuration of Nginx for HTTPS termination. The provisioning scripts are included in the Stonecutter repository, and allow for predictable reprovisioning of environments.

The Stonecutter application and database are deployed in docker containers, which offer the benefits of encapsulating some package dependencies and managing processes by starting and stopping containers.

Some deployment scripts are included in the infrastructure code for the deployment and redeployment of builds of the Stonecutter application into a Java Docker container.

17. Database Design

Stonecutter stores data required for OAuth2 login in four document collections in MongoDB. The collections are schema-less and the data is stored as BSON (Binary JSON) documents. Below is a brief description of the shape of the records in each collection:

Clients

Registered clients are stored in the client collection. As well as the name, ID and client secret, the client's url is stored so that redirects can be validated to prevent client-spoofing attacks.

```
{"url" : <CLIENT_URL>, "client-secret" : <CLIENT_SECRET>,
 "client-id”: <CLIENT_ID>, "name" : <CLIENT_NAME>}
```

Users

The user record consists of their login (i.e. their e-mail address), a unique ID generated by Stonecutter, their hashed and salted password, and a list of the IDs of clients that the user has authorised to access their user details.

```
{"authorised-clients": [<CLIENT_1_ID>, <CLIENT_2_ID>, etc..  ],
 "login" : <USER_EMAIL>, "password" : <PASSWORD>,
 "uid" : <USER_ID>}
```

Authorisation Codes

Authorisation code records have a reference to the user (subject) and client involved in an OAuth2 transaction, as well as the authorisation code itself, and the redirect-uri that the auth code is delivered to. These records are short-lived, and removed as soon as the access token is generated.

```
{"auth-code": <AUTH_CODE>
 “subject”:<USER>
 “client”:<CLIENT>
 “redirect-uri”:<REDIRECT_URI>}
```

Tokens

The token collection is used to store access tokens and temporary session tokens. Each token is associated with the corresponding client and user (i.e. subject).

```
{"subject" :<USER>,
 "client" : <CLIENT>,
 "token" : <TOKEN_ID>}
```

18. OAuth2

1. The client application redirects the user to the URL below:
   https://sso.dcentproject.eu/authorisation?response_type=code&client_id=CLIENT_ID&redirect_uri=CALLBACK_URL
   The CLIENT_ID is the unique ID for the client application registered with Stonecutter. The CALLBACK_URL is the URL that the server will redirect the user to, to initiate the next part of the sequence.

2. Once the user has signed in to the server, the server redirects the user to the URL below, with an authorisation code:
   https://client-application.org/callback?code=AUTHORISATION_CODE

3. The client's callback endpoint triggers an HTTP POST to
   https://sso.dcentproject.eu/oauth/token?client_id=CLIENT_ID&client_secret=CLIENT_SECRET&grant_type=authorization_code&code=AUTHORIZATION_CODE&redirect_uri=CALLBACK_URL
   The CLIENT_SECRET is the secret shared with the client application when it is registered with Stonecutter. The client also responds authorisation code sent by Stonecutter in the preceding redirect. For demonstration purposes the client ID and secret have been added as query parameters to the above URL. In practise the client ID and secret should be base64 encoded and added to the request header under an "authorization" key in the format CLIENT_ID:CLIENT_SECRET.

4. If the interaction is successful Stonecutter will respond to the HTTP request with a response that has a JSON body, with the access token and additional user details.

19. Testing

Test Driven Development has been used to develop Stonecutter features. Unit tests are written in advance of implementation, to ensure that the subsequent implementation is complete and of high-quality. The tests also provide invaluable documentation for open-source contributors looking to understand the function of the existing code, as well as confidence that new features do not alter existing functionality.

The Stonecutter development team has adopted the practises of Continuous Integration (CI) and Continuous Delivery. Each new commit to the code base automatically triggers a run of the test-suite in a cloud-hosted build pipeline. In addition, automated, browser-driven integration tests are triggered, to ensure successful integration between Stonecutter and a test client application. This ensures that the API contract between Stonecutter and integrating client applications is not broken.

Each new commit is also automatically deployed to a public-facing 'staging' environment so that the candidate build can be inspected before release. Releases to 'production' (the pilot deployment at https://sso.dcentproject.eu) are made multiple times per day.

Ad-hoc manual penetration testing has also been performed on the deployed instance of Stonecutter.

20. In Development

Some of the outstanding features that have been committed to will be built alongside development of the Secure Notifications tool (D5.6). In particular, administration and user access controls will allow organisations to ensure that secure notifications are delivered to trusted users. Developing these two feature sets in parallel will confirm that the requirements of the Secure Notifications tool are properly met by Stonecutter.

The following are details of Stonecutter features currently being analysed and developed:

20.1 Identity Management / Open ID Connect

Open ID Connect

- The OAuth2 implementation of Stonecutter will be modified to support the Open ID Connect standard, in addition to 'regular' OAuth2. In particular the Json Web Token (JWT) standard will be used for returning user profile details if Open ID interaction is requested by the client application.

Additional user details

- Users will be able to add additional personal information to their Stonecutter profiles (e.g. a biography, a site url).
- Users will be able to add a profile image to their account
- Users will be able to edit their personal information
- Clients will be able to request tailored subsets of the available user data to be provided in the token success response

20.2 Admin/individual user and group profiles

- An administrative account will be configurable when an instance of Stonecutter is deployed.
- The admin account will be able to view a list of registered users
- The admin account will be able to grant/revoke permissions to registered users
- Client applications will be informed of the permissions that a user has when the user is logged in

20.3 Groups and access control

- Administrators will be able to set coarse-grain access control for users to roles such as 'read-only' or 'read/write'. This information will be provided to client applications on request, so that client applications are able to make decisions about tool-specific permissions to grant users based on their Stonecutter role.

- Domain-specific group information relevant to integrated tools will be managed by those tools.

20.4 Other

Email notifications

- User will receive an email to confirm their email address upon registration
- If the user has forgotten their password, they will be able to request an email containing a link to reset their password
- If the user changes their password they will receive an email notification

App themes

- When an organisation deploys an instance of Stonecutter they will be able to configure the application with a theme, so that it is clear that the instance will fit with the organisation's branding

UI for client registration

- Admin users will be able to view a list of registered client applications
- Admin users will be able to register new client applications so that they are able to integrate with the Stonecutter instance
- Admin users will be able to unregister client applications so that they are no longer able to log in users through Stonecutter

21. Future Features

Some future possible developments that have been identified are:

- User profile attributes could also be presentable in the VCard format.
- Users could be able to download their profile attributes in a portable format
- The Secure Remote Password (SRP) protocol could be used to further strengthen password security.