Diploma Timeline

Claim Codes

> A web site that receives a claim code and renders a badge and diploma

Design

All pages

Must have:

- Mobile, tablet and desktop variants.
- A color palette functional to people with color blindness.

Landing page

This would be the root URL with an input for capturing an alphanumeric code.

It must have:

- A headline prompting visitor to introduce a claim code.
- A text input for code with validation feedback.
- A submit button.

It could optionally have:

- A notification widget for displaying form feedback.
- A button to enable the user to capture a QR code.

Preview Page

After the user introduces a code, they would be presented with this page.

It must have:

- A text input for the personalized name with validation feedback.
- A CTA to generate the final diploma.
- A preview of the diploma.

It could optionally have:

- A Google profile chip for users signed in to My Edvolution.
- The name field automatically populated from Google profile.
- A mechanism to change the name derived from the Google profile.

Success Page

It must have:

- A preview of the diploma.
- A CTA to download the diploma.
- A CTA to share the diploma.
- A CTA to add the diploma to My Edvolution.

Development

All pages

Must implement:

- Page title and metadata
- The designs conforming the constraints defined in HTML, CSS and JavaScript, including responsive variants.
- Internationalization (i18n) & localization (i10n), initially in Spanish and English.
- Accessibility standards i.e. WAI-ARIA for WCAG 2.1 AA conformance.
- Web performance best practices.
- Analytics for visitors and conversions.

Landing page

It must implement:

- Form validation and submission.
- Reading query parameters for `claim_code`.
- Redirecting to preview page.

It could optionally implement:

- OpenGraph metadata.
- QR code capture.
- Notifications for form events.

Preview page

It must implement:

- Fetching assertion data.
- Rendering PDF in a frame.
- Form validation and submission.
- Updating the PDF based on the form inputs.
- Redirecting to success page.

It could optionally implement:

- Google oAuth.
- Display oAuth profile.
- Fill form with oAuth profile.
- Allow changing the information derived from the oAuth profile.

Success page

It must implement:

- Baking the badge from the metadata and assertion.
- Generating a static page for the diploma and badge.
- Rendering the resulting PDF in a frame in the page.
- Allowing the PDF to be downloaded rather than opened within the browser.
- Sharing to Twitter, Facebook, LinkedIn and WhatsApp.
- OpenGraph metadata for social sharing embeds.

Sphingi

> The Badgetree backend aka Sphingi requires some updates for the new functionality

Development

It must have:

- Batch assertions not automatically baking badges on creation.
- A string `claim_code` property on the Assertion entity.
- A route to look-up Assertions by UUID.
- A boolean `claimed` property on the Assertion entity.
- A route to update Assertions claimed property.