Jonathan Edens

1. Story Analysis and Process Improvement

[Chat-GPT](https://chatgpt.com/share/6824899c-a7ec-800b-94ef-883d5b1fc2cd) was used to provide validation of my reasoning.

**Given the provided User Story, what works well?**

The story is simple and can be distilled into four components – signing in, profile access, information updates, and signing out. It also contains clear entry and exit points, which helps to define the extent of the session. Furthermore, it emphasises the imperative requirements of the User – for access to and updates of their profile to be secure and straightforward.

**What needs clarification or improvement?**

* The story should explicitly state the different paths for valid and invalid credentials, and thereby for authorised and unauthorised users.
* Any error messages due to incorrect or malformed credentials at login should be clearly defined within the specifications, rather than being open to interpretation.
* If a user account has a different risk profile e.g. high risk of fraud or previous unauthorised access attempts, do we enforce stricter controls such as MFA by default? This should be defined.
* The navigation pathway should be explicitly described i.e. Account > Profile Settings to remove any ambiguity.
* The allowed and disallowed phone formats should be defined. We should specify whether or not we accept country codes (e.g., +44 or 44), what the maximum permitted string length is, and which characters are allowed. We should also state what the error messaging should be for the phone number field, and for it to adhere with the ‘straightforward’ requirement of the original User Story.
* Can a user delete their recorded phone number without providing a replacement value? Is a null value for this field permitted?
* Does a user input their address manually, or is the site connected to a postcode search API?
* If a user provides a new address using a postcode search, can they edit it prior to saving? Or do they have to manually input their entire address, thereby increasing the risk of data input errors?
* Can a user edit an already saved address, for example, to add/edit a house or business name, or to include an additional address line?
* What are the mandatory fields for updating an address, and are they explicitly demarcated?
* Can a User delete an existing address record?
* If a user wishes to update both their phone number and address, does this require two POST/PUT API calls to store the data, or can they submit them via a single Save action?
* Clarification required around how we notify the user of the success/failure of the save attempt, to indicate if there was a 200 OK response or a 500 Server Error. The relevant messaging should be declared within the acceptance criteria.
* What constitutes an acceptable failure rate for saving, but would still be classed as ‘reliable’? This needs to be quantified.
* Do we maintain an audit log when profile data is updated, to record previous values, and when and by whom they were updated?
* Do we delete all session data on sign out (cookies, JWT) and is sign out effected automatically by being left idle for, say, 15 minutes?

**Underlying assumptions:**

* That the users’ experience level and the design of the website are such that the process of finding the correct menu, and entering/updating data is facile.
* That *any* of the described functionality is pre-existing. It is unclear as to whether the sign in, home, and profile pages already exist, or whether their development forms part of this feature request.
* That the security architecture of the site meets our requirements for data transport and authentication purposes.
* That there is an existing navigation link to the Profile section in the menu system, and that the menu is apparent to the user.
* That the Profile section of the site has already been built and has the required functionality to input phone numbers and addresses/perform address searches.
* That we have a clear set of validation rules for the entry of phone numbers and that the required format is apparent to the user.
* That ‘address management’ covers both manual and search-based approaches.

**Guidance for writing better user stories:**

The Story headline should be written in the format of “As a [User Type], I want (to) [Required Action], So that [Clearly Defined Outcome]”. This provides a clearly defined and unambiguous purpose for the Story, and helps to guide the required steps and acceptance criteria.

When writing acceptance criteria, they should be specific and measurable, to avoid ambiguity. For every action and behaviour, we should clearly define what success and failure look like to the end user. For example, rather than ‘Clear field labelling’ we should define what those labels are; rather than ‘Clear validation’ we should state what the user will see both before and after entering valid and invalid data. Similarly, instead of ‘Clear error guidance’ and ‘Clear session handling’ we should instead define exactly what they look like in terms of messaging and browser behaviour – for example, ‘Your email/password is incorrect’ and ‘all session tokens are cleared and invalidated upon logout.’ Avoiding the use of ambiguous and non-deterministic language such as ‘clear’, ‘flexible’, ‘helpful’, and ‘reliable’ – instead, using specific and definitive terms - is key.

To improve readability of the acceptance criteria and facilitate their understanding, they should be written using BDD. This ensures that the development criteria are mapped to the acceptance criteria in a readily understood format, and ensures we can readily identify all pathways that need to be covered during development and testing.

We should also consider whether or not any given user story covers too many actions, and whether it could be vertically sliced to make it more manageable. In this example, it would be reasonable to separate the sign in/sign out process into a discrete story, since this could be independently developed and tested. The process of navigating to the Profile could also be extracted into a separate story, so this one would only be focused on the information updates – thereby making it much more specific and measurable in terms of success and failure, and much less complex and intertwined.

**To Improve testability:**

To improve testing, we should define the validation rules around phone number input, and explicitly declare what the field labels and error messaging should be in each instance. This will facilitate the writing of test cases around the definitive list of permutations, and minimises the need for speculative interpretation.

We should also specify what the API/Integration endpoints and HTTP response codes are, for login, address validation, save actions, etc. This will enable us to test the API endpoints more extensively to ensure coverage of all possible results, and verify their correct implementation.

1. **Test Strategy Development**

**1.**

**Title:** Register a new account.

**Objective:** Investor is able to register a new account in the Downing portal using valid credentials.

**Title:** Register a new account with existing email address.

**Objective:** Investor is unable to register with an existing email address and is instead prompted to log in.

**Title:** Register a new account with a malformed email address.

**Objective:** Investor is unable to register without providing an email address in the correct format.

**Title:** Register a new account with an incorrectly formatted phone number.

**Objective:** Investor is unable to register with alpha characters in their phone number (e.g. ex).

**2.**

**As a** prospective Investor

**I want to** fully register an account with Downing

**So that** I can access their website

**Setup Requirements:**

No previous account details are required, data will be automatically generated during the test.

URL: <https://bonds-client-test.downinglabs.co.uk/>

**Functional Specifications:**

This test is a ‘happy’ path and is therefore expected to proceed to completion without any errors.

**Sign Up Screen:**

* User can input their Title, First Name, Last Name, Date of Birth, Email Address, and Phone Number.
* User can select an option from the ‘How did you hear about us?’ select list.
* User can accept the terms and conditions.
* User can select their Communication Preference.
* User can click the ‘Sign up’ button and proceed to the next page.

**Investor Type Screen:**

* User can select their investor type (High Net Worth).
* User can verify and input their annual income.
* User can verify and input their asset worth.
* User can input their full name into the signature box.
* User can open the calendar and **only** selecttoday’sdate.
* User can select their source of funds.
* User can check all four mandatory checkboxes.
* User can click the ‘Save’ button and proceed to the next page.

**Risk Test:**

* User cannot save and proceed until they complete the form.
* User can select the correct risk options to verify their understanding of their investments.
* User can click the ‘Save’ button and procced.

**Restricted Access:**

* User can see the Restricted Access page and is greeted by `Hello ${FirstName}`.
* User can log out.

**Expected Results:**

The User is able to register with novel details.

The User is able to complete all forms without any form errors.

The User is able to submit the forms without any errors.

The User is able to proceed through the registration process to completion.

**Success metrics:**

A new User account is created and is accessible via the frontend – accessible via Profile Settings.

An email is sent to the User address to confirm and verify their registration.

A database entry is created which contains the correct user details.

No errors or warnings are encountered during the registration process.