**Task 1**

1. “Forgot password” email is not being sent for the user account.

**Priority: Urgent**

The “Forgot password” email is not being sent, preventing existing users who have forgotten their passwords from accessing their accounts.It is a common human error to forget a password and tis should always be taken into consideration with any software that uses login forms to interact with its customers. It is a critical usability issue that locks users out of the accounts which makes it impossible for them to have account access. It can cause frustration on the user's part and some may even leave to competitors if this type of issue persists so resolving this issue is vital for access of users.

2. Website language switcher is not working.

**Priority: High**

Given that this is a subscription service catering to an international market, not being able to switch languages significantly can cause problematic for accessibility for a large portion of your potential user base. This directly impacts user experience and could prevent many international users from fully engaging with the website and subscribing.

If it were not for an international market the priority would be low as users will likely still be able to use the website in the default language. It may hinder accesibility for some but would not prevent basic functionality for everyone.

3. “About-us” footer links are redirecting to the homepage.

**Priority: Low**

Incorrect links are not desirable and should always be fixed,but they don't directly prevent users from subscribing or using the core functionality of the service, which is the bottom line. The "About Us" page is typically informational, giving a background about the company, and less critical to the immediate user journey of subscribing and using the service.. Users can likely still navigate to the "About Us" information through other means like through the main navigation.

4. During registration month date-of-birth dropdown box is only showing “November”.

**Priority: High**

During registration month date-of-birth dropdown box is only showing “November”: While users might eventually figure out a workaround (if possible), this is a major usability flaw during a critical onboarding process. It can lead to incorrect data entry, potentially causing issues later with date of birth features or compliance.This directly impacts the acquisition of new subscribers.

5. “Complete registration” button not working.

**Priority: Urgent**

"Complete registration" button not working: This is a a very urgent issue which needs to be addressed as soon as possible. If new users cannot finish signing up for the service, te renders the entire subscription model unusable for them. There is a risk of losing potential customers.

**Task 2**

Below I havelisted the considerations to be taken into account as part of the test planning process:

**Functionality & Requirements:**

Voucher Generation:

* Voucher Generation Logic: How are vouchers generated? (e.g., sequential, random, with specific patterns). Will they be generated using randomization for unique numbers, sequential generation and /or using specific patterns.
* Bulk Generation: Does the system support generating large batches of vouchers? Perfomance and scalanility will need to be tested here
* Customization: Can vouchers have specific attributes such as expiry dates, usage limits, daily limits, specific product, discount amounts, discount percentages, reward points system

Voucher Redemption:

* Redemption Validation: Testing how the system validates a voucher. This can be done by checking for existence, expiry, usage limits, applicability,and other types of validation tests
* Redemption Methods: Involves testing how the vouchers can be redeemed like online input, scanning a QR code, in-store entry, purchasing from a banking app.Also checking whether partial redemption is possible and having the balance remain on the customer's account
* Redemption Errors: How does the system handle invalid or expired vouchers? Clear, concise and informative error messages are important.

Voucher Management:

* Voucher Status Tracking: How is the status of a voucher tracked (e.g., active, redeemed, expired, cancelled)? We need to verify status updates.
* Voucher Cancellation/Invalidation: Does the system allow for the cancellation or invalidation of vouchers? Also need to test it's impact on voucher redemption
* Voucher Information : Can administrators easily search for and view details of specific vouchers? Testing search functionality and data accuracy is important.

Performance:

* Generation Speed: How quickly can vouchers be generated, especially in bulk? Load testing might be required.
* Redemption Speed: Tests how quickly vouchers are validated and processed during redemption. This under extreme loads
* System Responsiveness: Tests the responsive of the administrative interface for managing vouchers

Usability:

* User-Friendliness: Is the interface for managing vouchers intuitive and easy to use for administrators?
* Customer Redemption Process: Is the voucher redemption process seamless, clear and straightforward for customers?

Reliability:

* System Stability: Is the system stable and can it handle and maintainexpected loads without crashing or data loss?
* Data Integrity: Is the voucher data accurate and consistent throughout the system lifecycle?

Scalability:

* Load Testing: Can the system handle a significant increase in the number of vouchers generated and redeemed as the service grows?

Maintainability:

* Troubleshooting : Is there troubleshooting assistance available for users in the case of common errors and is this constantly updated?
* Language and Currency Support: If vouchers involve monetary value, does the system support multiple currencies? Consider whether the system supports various currencies and languages or whether it is specific only to the country's currency with various languages, and testing this.
* Date and Time Formats: Are date and time formats handled correctly for different locales?
* Which payment methods can be used to puchase vouchers?eg. eft, paypal, instant money, etc

Security:

* Voucher Code Security: Testing how secure the generated voucher codes are against fraud, guessing or manipulation? Penetration testing will be helpfull here
* Redemption Security: Is the redemption process secure to prevent fraudulent use?
* Data Security: How is sensitive voucher data protected.Personal details of users.Consideration of the POPI act and other leagal compliance implications.

**Task 3**

**What would I test about this API?**

Connectivity and Status Codes: I would want to make sure that the API endpoints are reachable and return the expected HTTP status codes (e.g 200 OK for successful requests).

Response Content: I would check whether the structure of the JSON response and the content of the message field.

Error Handling : i would keep this in mind the more complex the API structure becomes

**Why would I test this?**

I would test this for the following reasons:

* Ensuring functionality by confirming that the API is working as expected and that it delivers the intended response.
* Improving Reliability: Consistent testing contributes to a more stable and reliable API.
* Tests serve as a form of documentation which show the API is intended to be used and what responses are to be expected.
* As the API evolves and changes over time, tests help ensure that new changes don't break existing functionality.

**How would I test this?**

* I would use Postman using the following approach:
* Import the Collection
* Examine Endpoints
* Create Test Cases
* Send Requests: I would send requests to the API endpoints using Postman's request builder.
* Write Assertions
* Run Collections: I would organize related tests into collections
* Monitor and Iterate: I would continuously run tests as the API is developed and updated, iterating on the tests as needed.

**Description of process for setting up tests :**

* I launched my Postman application.I created a new collection and named it "Cat Act Ninja API Tests" to organize my tests.
* I created a New Request: For the first test, I clicked the "+" button to create a new request.
* I configured the Request by selecting"GET" as the request type.
* I entered the API endpoint URL: <https://documenter.getpostman.com/view/1946054/S11HvKSz>
* I saved the request named "Verify Successful Response".
* Navigated to the "Tests" Tab: In the request editor, I clicked on the "Tests" tab.
* I wrote Assertions (Test 1):
* I used the pm.test() function to define a test case with a descriptive name ("Status code is 200").
* Inside the test function, I used pm.response.to.have.status(200) to assert that the HTTP response status code should be 200.
* I added another pm.test() to check the response time using pm.expect(pm.response.responseTime).to.be.below(200). This helps monitor performance.
* Saved the Request: I saved the changes to the request.
* Repeated Steps 3-7 for Test 2:
* I created a new request (or duplicated the first one and modified it).
* I named it "Verify Response Message".
* In the "Tests" tab, I used pm.response.json() to parse the JSON response body.
* I then used pm.expect(responseJson.message).to.eql("Meow!") to assert that the message field in the JSON response is exactly "Meow!".
* I repeated Steps 3-7 for Test 3:
* I created a new request
* I named it "Verify Content Type Header".
* In the "Tests" tab, I used pm.response.to.have.header("Content-Type", "application/json") to ensure the API is correctly setting the Content-Type header to indicate a JSON response.
* Ran the Tests: Finally, I ran the entire collection to see if the tests pass or fail. Postman provides clear feedback on the test results which I have attached