unipro

Project:	**UUID Issue remediation**

Title:	**Unipro Test Plan**

Compiled by:	Dan Munn, Deborah Dean
Reference:	

Microsoft CERTIFIED Partner

1. Project Details

Project Name:	UUID Issue Remediation
Project No:	
Project Manager:	Deborah Dean
Our Project Reference:	UUID Issue Remediation
Customer Name:	Pfizer
Customer Project Reference:	
Customer Project Manager:	

Signoff

NAME	Signature	Date
Pfizer Engineering Lead	Matthew Saunders	
Programme Manager	Shaun Froome	
Lead Developer	Dan Munn	

2. Distribution

NAME	Copied (tick)	Date

3. Revision History

Version	Author	Date	Revisions
1.0	Deborah Dean	29/9/16	

4. Appendices

Enter appendices in the following index and follow with appendices.

Appendix	Appendix Name
https://docs.google.com/spreadsheets/d/18zI-0lz2A_QsFm8j08-VDu_Gj0WG2jP8DhjbjRmQPuU/edit#gid=0	Unipro Project Plan

5. Contents

6. Introduction

6.1. Test Plan Objectives

To specify the testing procedure and verification evidence of applying the patches which fix the duplication of UUIDs across HCP Portals. It is intended for use by project personnel in understanding and carrying out prescribed test activities through successful completion.

The plan also covers the regression testing of self-service on environments that have been updated with the patches.

This test plan will be used by Unipro during the POC phase (see section 4.2) and any updates will be made before it is handed over to the Pfizer support team to use for the roll-out.

7. Scope

The scope of testing covers the UUID patches and the self-service regression testing of upgraded sites.

8. Out of scope

The following activities are running in parallel to the UUID patch fixes but are NOT part of this testing activity:

1. Regression testing of the HCP version changes - the patches will be rolled out onto each HCP Portal site as part of a new portal release strategy, however this document does NOT cover any regression testing of any part of the release upgrade apart from the UUID data fixes.
2. Behat testing
3. UAT
4. Browser compatibility Testing
5. Xxcx???????

9. Test Strategy

9.1. Acceptance Criteria

The roll-out of the UUID fix patches is deemed to be successful when it meets the following criteria:

1. POC is completed – for details of what is covered in the POC see section 4.2
2. Results of the POC are approved by the following colleagues and approval is given for the Pfizer support team to commence with the roll-out:
   a. Matthew Saunders
   b. Chad de Groot
   c. Tim Holt
   d. Scott Gavin

e. Alok ?

9.2. POC testing

To fix the duplicate UUID issues on the HCP Portal sites 3 patches have been written which need to be applied. The 3 patches are:

1. A patch that was based on Ankur's patch which fixes any duplicate UUIDs by assigning unique IDs.
2. An update to the panels module
3. An update to pfizer_hcp2_deploy

In order to confirm that the patches fix any duplicate UUIDs we need to take evidence of the existing issues on each environment before the patches are applied, and take evidence of the data validation afterwards.

There are 3 phases of testing that is required here:

1. Unit tests - We created a series of SQL scripts that will be run before the patches are applied and afterwards, and logged the evidence for comparisons before and after.
2. Regression testing of self-service (deploy)
3. Drupal unit tests

9.2.1. Unit testing

The purpose of these test cases is to validate that the patches that have been written to fix the duplicate UUIDs problem in HCP Portal is successful in fixing the data issues.

To do this we will be running a series of SQL scripts before the patches are applied to a site to capture the volume of duplicate UUIDs that currently exist and run the same scripts again after the patches have been applied.

The scripts will be run on dev3, stage and prod of the DK, PT and IT sites.

The results of both the before and after testing will be logged on a test evidence excel sheet called 'UUID duplication testing results.xls'.

It is unknown before the SQL scripts are run on each site what the volume of duplicate UUIDs is, but the successful outcome would be that there are 0 duplicate UUIDs after the patches have been applied.

NB
Before = The environment as it sits before any platform changes
After = After upgrading the environment fully to 2.6.15 / 2.7.7 / 2.8.1 (dependent on market current version) which includes the UUID patches. See xxxxxxx for list of sites and the release version they are on.

9.2.1.1. Prerequisites

The following are pre-requisite steps that need to happen both on the existing dev3, stage and prod sites before and again after the site is upgraded.

1. Create a new backup of the market dev3 database.
2. Create a new backup of the market stage database.
3. Create a new backup of the market production database.
4. Download the database in step 1 to env_dev3.sql.gz
5. Download the database in step 2 to env_stage.sql.gz
6. Download the database in step 3 to env_prod.sql.gz
7. Delete the local database env_dev3 if it exists.
8. Delete the local database env_stage if it exists.
9. Delete the local database env_prod if it exists.
10. Create the local database env_dev3.
11. Create the local database env_stage.
12. Create the local database env_prod.
13. Import the env_dev3.sql.gz into the local database env_dev3.
14. Import the env_stage.sql.gz into the local database env_stage.
15. Import the env_prod.sql.gz into the local database env_prod.

9.2.1.2. Unit testing test scripts

Below are the test scripts that need to be executed on the dev3, stage and production environment bother before and after the site has been updated.

These tests are logged in the Unipro Spira platform, ref : Project: HCP Duplicate UUIDs PR201.

Task: Number of UUIDs that are duplicated.
Description: Retrieve a number of UUIDS that have more than one display associated with them.
Passing criteria: The number returned by the query after the upgrade will show 0. Any other number than 0 after upgrade is considered a failure.
Execution requirement: On env_dev3 env_stage and env_prod databases.
Query:

```
SELECT COUNT(uuid) FROM (
        SELECT uuid, COUNT(uuid) FROM `panelizer_entity`
        INNER JOIN `panels_display` ON `panelizer_entity`.did = `panels_display`.did
        GROUP BY uuid HAVING COUNT(uuid) > 1
) uuid_count
```

Task: Number of displays that are impacted by UUID duplication.
Description: Retrieve a complete count of panels displays (potential revisions of content) that are impacted by duplicate UUIDs.
Passing criteria: The number returned by the query after the upgrade will show 0. Any other number than 0 after upgrade is considered a failure.

Execution requirement: On env_dev3 env_stage and env_prod databases.
Query:

```
SELECT SUM(cuuid) FROM (
        SELECT uuid, COUNT(uuid) cuuid FROM `panelizer_entity`
        INNER JOIN `panels_display` ON `panelizer_entity`.did = `panels_display`.did
        GROUP BY uuid HAVING COUNT(uuid) > 1
) uuid_count
```

Task: Total number of customised displays.
Description: Retrieve a complete count of nodes (mindful of paywall etc customisation) that have customisation against it, including customisations by view-mode.
Passing criteria: The number returned by the query after the upgrade will be identical to the value collected before the upgrade.
Execution requirement: On env_dev3 env_stage and env_prod databases.
Query:

```
SELECT COUNT(*) FROM `node`
INNER JOIN `panelizer_entity` ON (`node`.`nid` = `panelizer_entity`.`entity_id` AND
`node`.`vid` = `panelizer_entity`.`revision_id` AND `panelizer_entity`.entity_type =
'node')
INNER JOIN `panels_display` ON (`panelizer_entity`.did = `panels_display`.did)
```

Task: Total number of customised pages.
Description: Total number of nodes that have customisation on them irrespective of view-mode.
Passing criteria: The number returned by the query after the upgrade will be identical to the value collected before the upgrade.
Execution requirement: On env_dev3 env_stage and env_prod databases.
Query:

```
SELECT COUNT(DISTINCT nid) FROM `node`
INNER JOIN `panelizer_entity` ON (`node`.`nid` = `panelizer_entity`.`entity_id` AND
`node`.`vid` = `panelizer_entity`.`revision_id` AND `panelizer_entity`.entity_type =
'node')
INNER JOIN `panels_display` ON (`panelizer_entity`.did = `panels_display`.did)
```

Task: Total number of nodes affected by UUID duplication.
Description: Total number of nodes that have customisation on them that is impacted by UUID duplication anywhere within history.
Passing criteria: The number returned by the query after the upgrade will show 0. Any other number than 0 after upgrade is considered a failure.
Execution requirement: On env_dev3 env_stage and env_prod databases.
Query:

```
SELECT COUNT(distinct node.nid) FROM `node`
INNER JOIN `panelizer_entity` ON (`node`.`nid` = `panelizer_entity`.`entity_id` AND
`node`.`vid` = `panelizer_entity`.`revision_id` AND `panelizer_entity`.entity_type =
'node')
INNER JOIN `panels_display` ON (`panelizer_entity`.did = `panels_display`.did) WHERE
`panels_display`.`uuid` IN (
        SELECT uuid FROM `panelizer_entity`
        INNER JOIN `panels_display` ON `panelizer_entity`.did = `panels_display`.did
GROUP BY uuid HAVING COUNT(uuid) > 1)
```

Task: Total number of pages with any collision vector.

Description: Total number of displays that have any collision vector (stealth or visible) between stage and production.
Passing criteria: The number returned by the query after the upgrade will show 0. Any other number than 0 after upgrade is considered a failure.
Execution requirement: On env_stage database (note it will cross-db query env_prod).
Query:

```
SELECT COUNT(distinct node.nid) FROM `node`
INNER JOIN `panelizer_entity` ON (`node`.`nid` = `panelizer_entity`.`entity_id` AND
`node`.`vid` = `panelizer_entity`.`revision_id` AND `panelizer_entity`.entity_type =
'node')
INNER JOIN `panels_display` ON (`panelizer_entity`.did = `panels_display`.did)
WHERE `panels_display`.`uuid` IN (
        SELECT uuid FROM `env_prod`.`panelizer_entity`
        INNER JOIN `env_prod`.`panels_display` ON `env_prod`.`panelizer_entity`.did =
`env_prod`.`panels_display`.did GROUP BY uuid HAVING COUNT(uuid) > 1)
```

Task: Total number of pages with a potential visual collision vector.
Description: Total number of displays that have any collision vector that mean there is definitive uncertainty (n-1 / n) that it will cause a visual defect if deployed as is.
Passing criteria: The number returned by the query after the upgrade will show 0. Any other number than 0 after upgrade is considered a failure.
Execution requirement: On env_stage database (note it will cross-db query env_prod).
Query:

```
SELECT COUNT(distinct node.nid) FROM `node`
INNER JOIN `panelizer_entity` ON (`node`.`nid` = `panelizer_entity`.`entity_id` AND
`node`.`vid` = `panelizer_entity`.`revision_id` AND `panelizer_entity`.entity_type =
'node')
INNER JOIN `panels_display` ON (`panelizer_entity`.did = `panels_display`.did) WHERE
`panels_display`.`uuid` IN (
        SELECT `env_prod`.`panels_display`.`uuid` FROM `env_prod`.`node`
        INNER JOIN `env_prod`.`panelizer_entity` ON (`env_prod`.`node`.`nid` =
`env_prod`.`panelizer_entity`.`entity_id` AND `env_prod`.`node`.`vid` =
`env_prod`.`panelizer_entity`.`revision_id` AND `env_prod`.`panelizer_entity`.entity_type
= 'node')
        INNER JOIN `env_prod`.`panels_display` ON `env_prod`.`panelizer_entity`.did =
`env_prod`.`panels_display`.did
        GROUP BY `env_prod`.`panels_display`.`uuid`
        HAVING COUNT(`env_prod`.`panels_display`.`uuid`) > 1)
```

9.2.1.3. *Local testing*

Before the scripts above were applied to the Acquia environments we downloaded several databases that we could tests the scripts on, to make sure they did the job intended.

The sites we tested on locally (using dev3, stage and prod) were:

Italy – v 2.6
France – v 2.6
Turkey – v 2.6
Netherlands – v 2.6
Russia – 2.6
UK – v 2.7

Denmark – v 2.7

The results of these local tests can be seen on 'UUID duplication local testing results.xls'.

eg

Before update

	Dev3	Stage	Prod
UUIDs that are duplicated	20	11	11
UUID count (duplicated displays)	1687	992	1651
TOTAL custom panelized PAGES	171	189	161
TOTAL custom panelizer nodes	169	187	159
TOTAL custom displays	151	119	149
Total likely UUID affected nodes	157	119	155

	Dev3 to Stage	Stage to Prod
TOTAL DEFINITE COLLISIONS	151	117
TOTAL DEFINITE VISUAL COLLISIONS	147	114

After update

	Dev3	Stage	Prod
UUIDs that are duplicated	0	0	0
UUID count (duplicated displays)	0	0	0
TOTAL custom panelized PAGES	171	189	160
TOTAL custom panelizer nodes	169	187	158
TOTAL custom displays	0	0	0
Total likely UUID affected nodes	0	0	0

	Dev3 to Stage	Stage to Prod
TOTAL DEFINITE COLLISIONS	0	0
TOTAL DEFINITE VISUAL COLLISIONS	0	0

(Above snippet taken from 'UUID duplication local testing results.xls')

9.2.2. Self service deploy testing post upgrade

After successful results of the POC unit testing above, we will manually test some sites post-platform upgrade, i.e. after upgrading the environment fully to 2.6.15 / 2.7.7 / 2.8.1 (dependent on market current version) which includes the UUID patches. Go to this link to the platform inspector for a list of sites and the release version they are on https://pi.digitalpfizer.com/adhoc-hcp-portal-version.

The reason we are doing this self-service testing, i.e. testing deploy, is that the duplicate UUIDs were being created during the deploy process itself.

Note can't do this self service testing on the same environments used for the POC unit testing as we can't do destructive testing on those POC environments, i.e. to carry out the self-service testing we need to create and delete new content.

Instead, we are using identical environments and testing them locally (i.e. not on the Acquia platform). We did this by downloading the environments (as listed below) and applying the code of the release for that version to the local sites.

How are we ensuring these are identical environments?

The aim was to conduct this testing before the POC unit testing was done – however although the patches were available the platform updates had not been implemented, so we'll have to do this testing in parallel with the POC Unit testing.

Risk: xxxxxxx
The environments we are using for this self-service testing are :

UUID Testing - Clone of US DEV3	pfprofessional2comdev.prod.acquia-sites.com
UUID Testing - Clone of US STAGE	pfprofessional2comdev2.prod.acquia-sites.com
UUID Testing - Clone of DK DEV3	pfprofessional2comdev3.prod.acquia-sites.com
UUID Testing - Clone of DK STAGE	pfprofessional2comdev4.prod.acquia-sites.com
UUID Testing - Clone of FR DEV3	pfprofessional2comdev5.prod.acquia-sites.com
UUID Testing - Clone of FR STAGE	pfprofessional2comdev6.prod.acquia-sites.com
UUID Testing - Clone of UK DEV3	pfprofessional2comdev7.prod.acquia-sites.com
UUID Testing - Clone of US STAGE	pfprofessional2comdev8.prod.acquia-sites.com

9.2.2.1. Prerequisites

Before testing can commence we need the following prerequisites:

- A version of the code that can be deployed to the website; this may not be a 'proper' new version because the existing versions are being re-worked at the moment. It may need to be deployed as overrides.
- A list of module versions to test with (see test 1).
- SQL to run against the DBs to validate that there are no duplicates (see test 2).
- One or more pairs of test servers (called DEV3 and STAGE in this document). These need to be configured to allow deploying of data from DEV3 to STAGE.

9.2.2.2. Risks

The following risks have been identified:

- The UK DEV3 server's database is broken (it has 2.7.7 code in it, and has been reverted to 2.7.5). We cannot effectively test on the UK site.

- Not all changes have been implemented yet.
- The new versioning system is not ready yet.
- The majority of changes are to Contrib or Pfizer controlled modules that are beyond our ability to directly control.
- Some change to Contrib modules will be quite extensive.

9.2.2.3. Users

- The test scripts refer to the following users:
    - Admin – this is 'User 1' and has full admin rights on the server
    - Content Editor – this is a user with the 'Content Editor' role, if running on a 'Self Service 2.0' system, this user will have the role 'Self Service Editor'.

9.2.2.4. Self-service regression testing test scripts

Below are the test scripts that need to be executed on the dev3, stage and production environment bother before and after the site has been updated.

These tests are logged in the Unipro Spira platform, ref : Project: HCP Duplicate UUIDs PR201.

After every test where possible (TBC) but certainly after every content type test is complete we will run the SQL queries, in order to capture any issues as soon as possible, rather than waiting to the end of the complete suite of self-service regression testing listed below.

PANELIZED NODE TESTS:

1. Verify that correct version of modules are installed
    a. Steps
        i. On ALL servers
        ii. Log in as Admin
        iii. Browse to module list (/admin/modules)
        iv. Check module versions (versions TBC)
    b. Success criteria
        i. Module versions are correct (versions TBC)

2. Verify that deploying the latest code has fixed any existing data issues
    a. Steps
        i. Obtain a copy of ALL databases (you may need to contact BT to do this)
        ii. Run the SQL (TBC)
    b. Success criteria
        i. The SQL reports that there are no duplicates.

3. Ensure that the Deploy Queues have been emptied.
    a. Steps
        i. On the DEV3 Server
        ii. Log in as Admin
        iii. Browse to the list of deploy queues (/admin/structure/deploy)
        iv. For each queue,

            1. If the queue does not state ‘Currently no content in this plan.’ then click the ‘Empty’ button for the queue to empty it.
            2. On the confirmation page, click ‘Empty’
   b. Success criteria
      i. All queues are now empty.

NOTE – From now on in the document I will use the term ‘Empty Deploy Queues’ – to do so, log in as the required used on DEV3 and follow the instructions above.

4. Ensure that the ‘Push to Next’ queue works – create sample content.
   a. Steps
      i. On the DEV3 Server
      ii. Log in as Admin
      iii. On the content page (/admin/content) click ‘Add content’
      iv. Select ‘HCP Article’
      v. On the new node, set the following values (warning, field names may vary if translated):
         1. Title: “Test 4 Ensure that the ‘Push to Next’ queue works”
         2. Principle Content: “Test 4 Ensure that the ‘Push to Next’ queue works”
         3. Branding: Select a valid Brand and Indication OR Medical Condition, the exact value is of no importance as long as it can pass validation.
         4. Paywall State: Public
         5. IM Fields: Primary Message: “Test 4 Ensure that the ‘Push to Next’ queue works”
         6. IM Fields: Primary Message Category: Adverse events
         7. IM Fields: Web Asset Type: Landing Page
         8. Published: Unchecked
      vi. Click ‘Save’
   b. Success criteria
      i. Your new item appears in the list of ‘Content’

NOTE – From now on in the document I will use the term ‘Create Content’ – to do so, log in as the required used on DEV3 and follow the instructions above to add a new ‘HCP Article’, but selecting a different content type if required. If a content type requires specific fields to be set, please populate with sample data.

5. Ensure that the ‘Push to Next’ queue works – deploy content.
   a. Steps
      i. On the DEV3 Server
      ii. Log in as Admin
      iii. On the deploy page (/admin/structure/deploy)
      iv. Find the ‘Push to Next’ queue
      v. Assuming the new node from 4 is in the queue, click ‘Deploy’
      vi. On the confirmation page, press ‘Deploy’
   b. Success criteria
      i. Your new item appears in ‘Push to Next’ deploy queue.
      ii. The ‘Deploy’ does not report any error when you click the button.

NOTE – From now on in the document I will use the term ‘Deploy Content’ – to do so, log in as the required used on DEV3 and follow the instructions above.

6. Ensure that the ‘Push to Next’ queue works – validate deployed content.

    a. Steps
        i. On the STAGE Server
        ii. Log in as Admin
        iii. Browse to the content page (/admin/content)
        iv. Edit the content called "Test 4 Ensure that the 'Push to Next' queue works"
    b. Success criteria
        i. Content item "Test 4 Ensure that the 'Push to Next' queue works" exists on STAGE
        ii. The values for each of the fields in "Test 4 Ensure that the 'Push to Next' queue works" match those on DEV3

NOTE – From now on in the document I will use the term 'Validate Deployed Content' – to do so, log in as the required used on STAGE and follow the instructions above.

7. Add Content – HCP Article
    a. Steps:
        i. As Admin:
            1. Empty Deploy Queues
        ii. As Content Editor:
            1. Log into DEV3
            2. Create Content – HCP Article
                a. Title: "Test 7: Add Content – HCP Article"
                b. Principle Content: "Test 7: Add Content – HCP Article"
                c. Branding: Anything valid.
                d. Paywall: Public
                e. IM Fields: Anything valid.
                f. Published: Yes
            3. Deploy Content
            4. Validate Content
    b. Success criteria
        i. Deployed content is valid

8. Edit Content – HCP Article
    a. Steps:
        i. As Admin:
            1. Empty Deploy Queues
        ii. As Content Editor:
            1. Log into DEV3
            2. Edit the content called Test 7: Add Content – HCP Article
                a. Title: "Test 7: Add Content – HCP Article Edited"
                b. Paywall: Private
            3. Deploy Content
            4. Validate Content
    b. Success criteria
        i. Deployed content is valid

9. Edit Panelized Content – HCP Article
    a. Steps:
        i. As Admin:
            1. Empty Deploy Queues

ii. As Content Editor:
1. Log into DEV3
2. View the content called Test 7: Add Content – HCP Article Edited
3. In the IPE, click 'Customize this page'
4. On a Panelized Row, click the "+" button.
5. In the 'Add content to…' popup, add an existing bean to the page.
6. Click 'Save' in the popup
7. Ensure that the 'Create new revision' checkbox is checked in the IPE
8. Add a revision message in the 'Log Message' box of the IPE
9. Click 'Save as custom' in the IPE
10. Deploy Content
11. Validate Content

b. Success criteria
i. Deployed content is valid

10. Edit Panelized Paywall Content – HCP Article
a. Steps:
i. As Admin:
1. Empty Deploy Queues
ii. As Content Editor:
1. Log into DEV3
2. Edit the content called Test 7: Add Content – HCP Article Edited
a. Paywall: Paywall
3. View the content called Test 7: Add Content – HCP Article Edited
4. In the IPE, click 'View Mode'
5. Select 'Paywall'
6. In the IPE, click 'Customize this page'
7. On a Panelized Row, click the "+" button.
8. In the 'Add content to…' popup, add an existing bean to the page.
9. Click 'Save' in the popup
10. Ensure that the 'Create new revision' checkbox is checked in the IPE
11. Add a revision message in the 'Log Message' box of the IPE
12. Click 'Save as custom' in the IPE
13. Deploy Content
14. Validate Content

b. Success criteria
i. Deployed content is valid

11. Edit Panelized Paywall Layout – HCP Article
a. Steps:
i. As Admin:
1. Empty Deploy Queues
ii. As Content Editor:

1. Log into DEV3
2. View the content called Test 7: Add Content – HCP Article Edited
3. In the IPE, click 'Change layout'
4. From the category, select 'HCP Universal'
5. Select a new layout (e.g. 'HCP: Universal grid')
6. Fill in any required values (e.g. Number of rows: 5)
7. Click 'Save as custom'
8. In the IPE, click 'Customize this page'
9. On a Panelized Row, click the "+" button.
10. In the 'Add content to…' popup, add an existing bean to the page.
11. Click 'Save' in the popup
12. Ensure that the 'Create new revision' checkbox is checked in the IPE
13. Add a revision message in the 'Log Message' box of the IPE
14. Click 'Save as custom' in the IPE
15. Deploy Content
16. Validate Content

b. Success criteria
   i. Deployed content is valid

12. Edit Panelized Paywall New Bean – HCP Article
   a. Steps:
      i. As Admin:
         1. Empty Deploy Queues
      ii. As Content Editor:
         1. Log into DEV3
         2. Browse to the Bean content page (/admin/content/blocks)
         3. Click 'Add block'
         4. Select a block type (as you repeat this test, try different bean types.) e.g. 'HCP: Simple Bean'
         5. Provide some sample data for the bean, ensure you label it with a value you can remember.
         6. View the content called Test 7: Add Content – HCP Article Edited
         7. In the IPE, click 'Customize this page'
         8. On a Panelized Row, click the "+" button.
         9. In the 'Add content to…' popup, add your new bean to the page
         10. Click 'Save' in the popup
         11. Ensure that the 'Create new revision' checkbox is checked in the IPE
         12. Add a revision message in the 'Log Message' box of the IPE
         13. Click 'Save as custom' in the IPE
         14. Deploy Content
         15. Validate Content
   b. Success criteria
      i. Deployed content is valid (including new bean)

13. Edit Panelized Paywall Move and Delete (multiple revisions) – HCP Article

a. Steps:
    i. As Admin:
        1. Empty Deploy Queues
    ii. As Content Editor:
        1. Log into DEV3
        2. View the content called Test 7: Add Content – HCP Article Edited
        3. In the IPE, click 'Customize this page'
        4. Drag and drop panels from one row to another.
        5. Ensure that the 'Create new revision' checkbox is checked in the IPE
        6. Add a revision message in the 'Log Message' box of the IPE
        7. In the IPE, click 'Customize this page'
        8. Delete a panel
        9. Click 'Save as custom' in the IPE
        10. Ensure that the 'Create new revision' checkbox is checked in the IPE
        11. Add a revision message in the 'Log Message' box of the IPEDeploy Content
        12. Validate Content
b. Success criteria
    i. Deployed content is valid (including new bean)

REPEAT TESTS FOR PANELIZED PAGES:

14. Test Content Type – HCP Events: Live Event
    a. Steps: Repeat Tests 7 through 14 using the content type 'HCP Events: Live Event'

15. Test Content Type – HCP Indications
    a. Steps: Repeat Tests 7 through 14 using the content type 'HCP Indications'

16. Test Content Type – HCP Landing Page
    a. Steps: Repeat Tests 7 through 14 using the content type 'HCP Landing Page'

17. Test Content Type – HCP Medical Condition
    a. Steps: Repeat Tests 7 through 14 using the content type 'HCP Medical Condition'

18. Test Content Type – HCP News Article (Maybe called Newscred Article, depending on version)
    a. Steps: Repeat Tests 7 through 14 using the content type 'HCP News Article'

19. Test Content Type – HCP Products
    a. Steps: Repeat Tests 7 through 14 using the content type 'HCP Products'

20. Test Content Type – HCP Support and Services
    a. Steps: Repeat Tests 7 through 14 using the content type 'HCP Support and Services'

21. Test Content Type – HCP Tabs Article
    a. Steps: Repeat Tests 7 through 14 using the content type 'HCP Tabs Article'

22. Test Content Type – HCP Tabs Support and Services
    a. Steps: Repeat Tests 7 through 14 using the content type 'HCP Tabs Support and Services'

TESTS FOR POLLS:

23. Test Content Type – HCP Poll: Customer Feedback
    a. Steps:
        i. As Admin:
            1. Empty Deploy Queues
        ii. As Content Editor:
            1. Log into DEV3
            2. Create Content – HCP Article
                a. Title: "Test 23: Add Content – Customer Feedback"
                b. Published: Yes
                c. Provide valid content.
            3. Deploy Content
            4. Validate Content
    b. Success criteria
        i. Deployed content is valid

24. Edit Panelized Content – HCP Article
    c. Steps:
        i. As Admin:
            1. Empty Deploy Queues
        ii. As Content Editor:
            1. Log into DEV3
            2. View an existing Article
            3. In the IPE, click 'Customize this page'
            4. On a Panelized Row, click the "+" button.
            5. In the 'Add content to…' popup, add the poll created in step 23
            6. Click 'Save' in the popup
            7. Click 'Save as custom' in the IPE
            8. Deploy Content
            9. Validate Content
    d. Success criteria
        i. Deployed content is valid

25. Test Content Type – HCP Poll: Quick Poll
    e. Steps: Repeat Tests 23 through 24 using the content type 'HCP Poll: Quick Poll'

TEST FOR SPECIALISED PANELIZED NODES:

26. Test Content Type – HCP Login
    a. Steps: Repeat Tests 7 through 9 using the content type 'HCP Login' – this content type does not have a concept of 'Paywall'.

TEST FOR NON PANELIZED NODES:

27. Test Content Type – HCP Basic Page
    a. Steps:
        i. As Admin:

                1. Empty Deploy Queues
            ii. As Content Editor:
                1. Log into DEV3
                2. Create Content – HCP Basic Page
                    a. Title: "Test 27: Add Content – Basic Page"
                    b. Published: Yes
                    c. Provide valid content.
                3. Deploy Content
                4. Validate Content
        b. Success criteria
            i. Deployed content is valid

28. Test Content Type – HCP Contact Us
    b. Steps: Repeat Test 1 using the content type 'HCP Contact Us'

29. Test Content Type – HCP Pre-event Survey
    c. Steps: Repeat Test 1 using the content type 'HCP Pre-event Survey'

30. Test Content Type – HCP Reference
    d. Steps: Repeat Test 1 using the content type 'HCP Reference'

31. Test Content Type – HCP: eDetail
    e. Steps: Repeat Test 1 using the content type 'HCP: eDetail'

32. Test Content Type – HCP: Live Webinar
    f. Steps: Repeat Test 1 using the content type 'HCP: Live Webinar'

33. Test Content Type – HCP: OnDemand Webinar
    g. Steps: Repeat Test 1 using the content type 'HCP: OnDemand Webinar'

POST TESTING VERIFICATION:

34. Verify that deploying the latest code has fixed any existing data issues
    a. Steps
        i. Obtain a copy of ALL databases (you may need to contact BT to do this)
        ii. Run the SQL (TBC)
    b. Success criteria
        i. The SQL reports that there are no duplicates.

9.2.3. Drupal unit tests

To provide additional validation some of the patches that have been generated include some unit/functional tests that allow automated validation of changes to the system.

There are a number of validation tests that have been added to the pfizer_hcp2_deploy module to ensure that we can verify how the deployment system behaves without having to interact and create data.

The test scenarios that are provided are:

1. Detection of UUID randomisation
   <Further information required>
2. Verification of UUID display uniqueness with the creation of content.
   <Further information required>
3. Ensure content deployment is rejected when it is new to the target and contains UUID tied to another content item
   <Further information required>
4. Ensure content deployment is rejected when it is existing but the UUID of its display belongs to another entity (or revision).
   <Further information required>
5. Content deployment is accepted when it is panelized, and contains a unique display UUID.
   <Further information required>
6. Content deployment is accepted when it is panelized, and contains a new UUID that is unique to the system.
   <Further information required>
7. Content deployment is accepted when it is panelized, and it is referencing its own display UUID.
   <Further information required>

9.3. White box testing

Based on what we know about the code and HCP, is there anything we can see that could be a risk? i.e. prod the code to see what happens?

9.4. Performance Testing

The patch is considered as having no impact on performance of the portal sites

- How do we know this? Can we prove it? Or get a 22nd opinion from Chad or Matthew??

9.5. Security Test

To assess any impact on security, a webscan will be run on the sites that are selected for the POC after the patch is applied. The outcome will determine if security scans needs to be run on each portal site that the patch is rolled out on.

The webscan reports will be filed with the other documentation for this project.

9.6. Visual regression testing

Whilst the patch applied corrects the placement of content where it was deployed onto a wrong page, this should not affect the display of content, however SmartCapture has been set up to take screenshots of the POC site before and after the patch implementation.

This is being led by Nate Swart. (ref https://docs.google.com/document/d/1nrzauZjnBlCtCK0TxuDewkGLkQGLFNpEMtlhsludTKs/edit#)

However, if the POC site screenshots show no visual differences it does not guarantee there will not be any on the rollout sites; instead its purpose is to highlight xxxxxxx??????

If visual defects are found,

9.7. Documentation Test

This testing document will be used to run through the testing steps during the POC and provide the testing evidence as stipulated in this document to validate this document before handing over to the support team to use during their roll-out of the patch.

This document will be approved for release to the support team by:

- Scott Gavin
- Matthew Saunders
- Danielle Hallett

9.8. Regression Testing

Full regression testing of the 2.8.2?? release of the HCP Portal will need to be done BEFORE the

we only need a full regression against the 2.8.x branch.

[2:39]
Ask Dan if he agrees if you wouldn't mind - but that's my thinking at this point. The 2.6 and 2.7 branches won't have accepted any major new functionality.

9.9. External and any third party integrations

Any other testing?

9.10. Test Limitations or Exclusions

10. Testing Risks

10.1. Test Plan Planning Risks and Contingencies

The following section details the risks to the project from the perspective of the test plan, and the mitigations / contingency introduced.

10.1.1. Availability of key personnel and testing resources

- Support availability to roll-out the patch
- Support onboarding and understanding of this document
- Skillset to understand and complete the testing

10.1.2. Local over-rides on specific modules

– ADD DETAILS FOR REMEDIATION – I.E. WE NEED TO STIPULATE WHICH V OF PANELISER NEEDS TO BE PRESENT (SHOULD THIS BE DOCUEMTNED IN THE SPIN UP GUIDE?)

10.1.3. Consequence of 2.6-2.7 merge project on markets

- At risk' markets will be removed from this process – TBC
- Define and document the process for at risk markets

10.1.4. Differences in the territory environments

10.1.5. Differences between each territory could be: xx and could cause the following issues to arise:

10.1.6. Roll-out on a market does not work

– Identify what might cause this to happen

– Mitigation to be documented

10.1.7. Late or non delivery of required hardware, software, test data or tools

10.1.8. Ability to test all features

10.1.9. Changes to the original requirements for testing

10.1.10. Solution not found by end of week

Solution -

10.1.11. What are the technical risks that could happen that would make us need to rollback? What is the rollback procedure? Are backups in place?

Eg will we need to test content has retained its paywall state?
Is the POC on Prod as well? Any risk there with live user data?

10.1.12. SmartCapture set up and run of before images delays roll-out

10.1.13. Webscan results on the 'after' scan shows defects caused by the patch

How to mitigate??

10.1.14. Unipro Support

Any support from the Unipro dev team to the Pfizer Support team during the roll-out may impact on other Pfizer projects

11. Test Schedule

Refer to the project plan here:

https://docs.google.com/spreadsheets/d/18zI-0lz2A_QsFm8j08-VDu_Gj0WG2jP8DhjbjRmQPuU/edit#gid=0

12. Test Documentation Deliverables

The top-level schedule for the test program is addressed in the test documentation deliverables. The test program schedule contains the major events, activities, and deliverables involved in the test program. Activities performed include the design, development, and execution of tests, and are described in the scheduled test documentation listed below.

- Test Plan: Test planning documentation
- Test Script(s) : The test scripts that detail what to test and the results of the test
- Test Summary Report: A report that includes the results of each test script, including summary details on unresolved defects identified during testing
- Test cases
- Test data and simulators
- Configured tools and their outputs
- Incident reports & Test issues log

13. Control Procedures

13.1. Reviews

Daily updates are in place during the Unipro testing on local environment through to the end of the POC.

Any amber or red issues will be flagged immediately to Scott Gavin.

13.2. Escalations from the support team during the rollout

Ankur to raise to the project team:

- Scott Gavin UK
- Lukas Kopac UK
- Matthew Saunders US
- Jim Farelly US
- Dan Munn UK

with the environment and details of the failure or issue faced.

13.3. Unipro support

Unipro will be on hand to provide support to the Pfizer support team during the roll-out of the patch to the HCP Portals

How long will this take???.

13.4. Change Request

If it is identified that the patch has failed on a specific site the project team will decide if a) the patch is modified (risks?) b) an extra patch is created

13.5. Defect Reporting

Defects will be

a) Duplicate UUIDs found after the patch has been applied to a site
b) Other issues not considered – the project team will need to decide if this defect is a side effect of the patch
c) Fix the issue regardless of the cause
    a. Unipro to fix if the patch has caused a previously unidentified defect
    b. Support team to fix if unrelated to the patch

14. Resources and Responsibilities

14.1. Resources

Resource	Who
Senior Project team	Scott Gavin Lukas Kopac Matthew Saunders Jim Farelly Dan Munn Ankur ??
Unipro development team	Dan Munn Simon Grimes Max Parker
Unipro PM	Deborah
Pfizer Support Team	(Led by) xxxxx

14.2. Responsibilities

Resource	Responsibilities
Senior Project team	• Make decisions on dealing with issues raised • Ensure Pfizer resource is in place and onboarded • Give approval for roll-out after POC • Create roll-out list for support team and deadlines
Unipro development team	• Local testing of patch and POC • Raise any issues to Unipro PM
Unipro PM	• Escalates any issues and new risks to the Senior Project team • Ensures testing documentation and spin up guide is approved appropriately and released on time as agreed with Scott Gavin.
Pfizer Support Team	• Rolls-out the patch • Escalates any issues and new risks to the Senior Project team • Ensures testing documentation released on time as agreed with Scott Gavin.

15. Deliverables

16. Suspension / Exit Criteria