Vardot Developers Cheat Sheet

by tareq_elayyan via cheatography.com/57649/cs/15261/

Site Building Best Practices

Plan your initial site: Always read the requirement specifications in detail. Clear any doubts so there is no assumptions about the functionality.
You should know what content type you are going to create in the project, how many views need to there, what is the layout of the site, how many fields need to be created and how everything should appear in responsive.

Compare Similar Modules instead of installing the first module you find.

Don't Use Too Many Modules

Don't Hack core or any contrib module or any contrib theme or any contrib profile. Patches could be used.

Don't create new field (base), if field already exist, Ex: field_image, field_new_image, field_event_image. Always try to reuse exist fields.

Don't Repeat Yourself (**DRY**): http://de-viq.com/don-t-repeat-yourself/

Naming: When you create a new block/-view, make sure that its name is human readable and understandable by the client.

Add Real Content for Testing

Always use **t()** function for strings; this is important for translations. And **l()** function to format any internal or external URL's.

Content Type Creation Guideline

Content Type and View Modes

1. Check content type name, description
2. Add the needed fields and configure them
3. Add the SEO and metatags fields
4. Configure the form display
5. Configure the view modes needed for the content type
6. Check content type Rabbit Hole settings (if it applies)
7. Check content type moderation settings (if it applies)

Entity Queues

8. Create an entity queue for the content type (if it is needed)
9. Configure the entity queue settings

Content Type Creation Guideline (cont)

Views

10. Create Listing view for the content type (Main View)
11. Create Home page view
12. Create view to show Most popular nodes
13. Create view to show most recent nodes
14. Execlude current node from both most recent and most popular views
15. Create view to display related nodes according to Taxonomy terms
16. Expose views Filter if needed

Node Page

17. Set the node page (Full content) view mode and style it.

Landing Page / Section Page

18. Create the content type Landing page / section page using page manager or landing pages

URL Aliases

19. Create Pathauto pattern for the content type
20. Create Pathauto pattern for the taxonomy terms

Metatags

21. Add and configure the content type metatags

Permissions

22. Set the view, edit and delete permissions for Site admin, Content admin and editor
23. Set the necessary permissions for SEO admin

Search Settings

24. Check search results and indexing settings

Definition of DONE (DONE is DONE)

Version Control (Git) Best Practices

1. Never push code to master branch, Code merge should be handled through pull requests
2. Never merge code to master branch before testing on your local machine.
3. Never push incomplete feature, functions, modules ....etc.
4. Developers should check git status (tracked and untracked files) before pushing code.
5. Git commit message is very important for history, and it's better to open task branch from it’s jira ticket.
6. Developers should check feature generated code before push the code
7. Don't include (files(sites/default/files), reports, IDEs directory, .swp, .tmp, .bak, ~.nib, .DS_Store, ...etc) in git repo.
8. Don't commit permissions to git (always ignore permissions "git config core.filemode false")
9. Always pull before committing new code to avoid conflicts, if you have a conflict solve on local then push the code.
10. Don't commit big images or files to git.

Implementing Atomic Design in Varbase

1. Install Varbase via composer.
2. Create a new Vartheme subtheme for your project. Steps Here
3. Install a style guide module. We are using Varbase Style Guide
4. Now you’ll be able to see all styles on /admin/appearance/styleguide
5. Edit (do not add) \YOUR_VARTHEME_-SUBTHEME\less\variables.less to match the design style guide. This should cover most of the atoms.
6. Vartheme is following smacss architecture in categorizing CSS rules.
7. Change (DO NOT ADD) needed components in \YOUR_VARTHEME_SUB-THEME\CATEGORY_DIR\X.less
8. Build \YOUR_VARTHEME_SUBTHEME-\base\your_theme.base.less to cover rest variations.
9. Do site building and test on real content.

We only consider development task is done if and only if:
1. Functionally tested for all possible scenarios
2. Tested on English
3. Tested on all other languages, if applicable
4. Tested on real demo content
5. Tested on Mobile (responsive)
6. Tested for browser compatibility
7. Users permissions were tested for view, edit and delete.
8. Includes “proper” and “final” copy/w-ording: Titles, descriptions, labels, messages, emails content/subject.
9. HTML Markup is Accessible.
10. CSS is atomic.

Task Technical Workflow

1. **Create a new (git) branch:** Go to the issue page on Jira and click on “Create branch” under "Development" in the right hand menu, page will redirect to bitbucket with fields already filled. Click on create branch then checkout the branch to your local.

2. **Implement changes on local:** Please make sure that all database changes can be deployed through files and managed through git.

3. **Push your changes to the development environment:** After you make sure the task is fully tested locally you will need to push it to the dev environment, this will require you to merge task branch into the development master/main branch through pull request.

Deployment Procedure

1. Schedule the release/deployment date (ONLY Sunday to Wednesday mornings). Deployments are not allowed on Thursdays unless an exception is provided.

2. A (git) tag should be created on the main branch to be used as a reference in the deployment.

3. DB Backup should be taken from the production environment.

4. Follow the steps in the Internal Deployment Checklist

5. The developer should fill the Deployment Request Form

6. A release should be created and configured on Jira.

7. Make sure all tickets that you want to release are included in the release and assigned to the same "fixVersion".

8. After the release is released, the PM will make sure to notify the client and release it on JIRA thereby closing all tickets.

9. Site or Work Package post-launch checklist should be filled the same day of the deployment.

Created by Vardot

Drupal Project Local Setup

1. Clone the project from BitBucket.

2. Download Drush aliases file and place it under your user ".drush" folder.

3. Copy example.settings.local.php into your site directory (ie. default) and change database credentials.

4. Create a database on your local machine and use the desired environment Drush command to sync DB.

5. Use ‘drush rsync’ to sync project files from server to your machine or use Stage File Proxy module in case files size is too big.

6. Create a virtual host for the project "project.local"

7. Run "drush cr" or truncate the cache tables in DB after you fully complete the above steps.

Developer's Checklists

[Site] Pre-execution Checklist	link
[WorkPackage] Pre-execution Checklist	link
[Site] Pre-launch Checklist	link
[WorkPackage] Pre-launch Checklist	link
[Site] Post-launch Checklist	link
[WorkPackage] Post-launch Checklist	link