Broadcasting Webservice for Smart TVs

Lennard Kittner, Henry Boos,
Jiangang Huang, Sicheng Dong,
Jannik Wibker

SMART TV
SYSTEM

Contents

1 Introduction

This document describes the testing phase of the smartTVSystem product. The main focus was on unit testing. Unit tests were used in all three parts of the project to discover bugs and ensure that all modules of the software function as intended. To make sure that not only the parts individually, but the whole system, is bug free we also performed manually the general test cases (GTC) from the requirements document to test the system as a whole. We also let other people use and explore the software to get feedback and find yet undiscovered bugs. The document contains all bugs that were found and all necessary code changes.

2 Test Line coverage

2.0.1 Frontend

Frontend Test coverage

Package	Testing phase		Implementation phase	
	stmts %	line %	stmts %	line %
All files	84.14	85.01	43.44	44.27
src	70	74.29	0	0
App.tsx	70	74.29	0	0
src/components	90.41	91.27	19.35	21.95
Announcement.tsx	100	100	77.78	100
CarouselSlot.tsx	100	100	0	0
EmptySlot.tsx	100	100	100	100
HorizontalSplit.tsx	100	100	0	0
Intl.tsx	100	100	100	100
Layout.tsx	87.5	89.29	0	0
VerticalSplit.tsx	100	100	0	0
WidgetSlot.tsx	77.78	77.78	62.5	62.5
src/plugins	82.76	80	17.39	20
index.ts	82.76	80	17.39	20
src/plugins/Cafeteria	86.67	88.37	21.74	22.73
Cafeteria.ts	100	100	100	100
CafeteriaStore.ts	100	100	60	60
CafeteriaWidget.tsx	81.25	83.33	3.03	3.23
src/plugins/Cafeteria/util	100	100	100	100
cafeteria.ts	100	100	100	100
src/plugins/Calendar	83.87	83.61	29.82	30.36
Calendar.ts	100	100	100	100
CalendarStore.ts	87.5	87.5	38.89	38.89
CalendarWidget.tsx	79.41	78.79	17.14	17.65
src/plugins/Calendar/util	100	100	89.47	100
events.ts	100	100	89.47	100
src/plugins/Clock	78.57	80.77	35.71	38.46

Clock.ts	100	100	100	100
ClockStore.ts	72.73	72.73	45.45	45.45
ClockWidget.tsx	78.57	83.33	14.29	16.67
src/plugins/Image	92.11	94.44	33.33	36.84
Image.ts	100	100	100	100
ImageStore.ts	94.44	94.44	50	50
ImageWidget.tsx	88.24	93.33	0	0
src/plugins/Publication	100	100	61.54	61.54
Publication.ts	100	100	100	100
PublicationStore.ts	100	100	71.43	71.43
PublicationWidget.tsx	100	100	0	0
src/stores	55.71	56.92	57.75	57.58
UIStore.ts	55.71	56.92	57.75	57.58
src/util	89.05	90.83	71.85	74.77
FileLoader.ts	80	86.36	66.67	72.73
api.ts	77.78	85.71	22.22	28.57
cachingResourceHandler.ts	100	100	94.59	100
check_type.ts	100	100	83.33	80
configHandler.ts	100	100	100	100
contexts.tsx	57.14	57.14	33.33	33.33
languageHandler.ts	83.33	100	83.33	100
load-stylesheet.ts	100	100	100	100
promise.ts	100	100	100	100
withToast.tsx	100	100	0	0

2.0.2 Dashboard

Dashboard Test coverage

Package	**Testing phase**		**Implementation phase**	
	stmts %	**line %**	**stmts %**	**line %**
All files	78.99	79.87	36.33	79.05
src	77.78	77.78	0	0
App.tsx	77.78	77.78	0	0
src/components	73.37	74.32	24.4	25.52
Button.tsx	91.67	91.67	83.33	83.33
ConfigFileSection.tsx	41.3	41.46	26.83	30.56
DatePicker.tsx	100	100	0	0
Intl.tsx	100	100	100	100
Login.tsx	88.89	100	0	0
PageWrapper.tsx	77.27	73.68	0	0
PrivateRoute.tsx	82.35	85.71	0	0
Table.tsx	94.74	93.33	84.21	80
TimeInput.tsx	81.82	85	0	0
src/pages	77.7	79.47	0	0
AnnouncementPage.tsx	83.64	86.54	0	0
CalendarPage.tsx	76.92	80.65	0	0
CoreConfigurationPage.tsx	90.91	88	0	0
ErrorLogPage.tsx	85.71	90	0	0
ImagePage.tsx	75.32	76.12	0	0
LoginPage.tsx	33.33	33.33	0	0
PluginConfigurationPage.tsx	87.5	91.67	0	0
index.tsx	52.17	52.63	0	0
src/stores	77.78	75	77.78	75
UIStore.ts	77.78	75	77.78	75
src/util	86.93	87.42	85.23	86.75
FileLoader.ts	78.57	85.71	78.57	85.71
check-type.ts	83.33	80	83.33	80
config-handler.ts	100	100	94.29	100
dates.ts	100	100	95.65	95.45
jwt.ts	88.89	87.5	88.89	87.5
jwtStrategy.ts	70.83	70.45	70.83	70.45
storage.ts	94.44	94.44	94.44	94.44
withToast.tsx	100	100	100	100

2.0.3 Backend

Backend Test coverage

Package	**Testing phase**		**Implementation phase**	
	class %	**line %**	**class %**	**line %**
com.pse.smartTVSystem	100	89	72	52
service	100	79	62	41
controller	100	96	20	3
repository	100	100	100	38
data	100	100	80	76
exception	100	100	100	100
com.pse.announcementPlugin	100	97	100	81
service	100	100	100	93
controller	100	95	100	50
repository	100	100	100	100
data	100	100	100	82
com.pse.cafeteriaPlugin	100	89	85	24
service	100	83	50	11
controller	100	100	100	35
data	100	100	100	49
com.pse.calendarPlugin	92	64	83	78
service	100	56	66	24
controller	100	96	100	61
repository	100	100	100	100
data	100	100	66	44
com.pse.imageDisplayPlugin	100	94	100	59
service	100	92	100	80
controller	100	92	100	52
repository	100	100	100	100
data	100	100	100	58
com.pse.publicationPlugin	100	77	75	32
service	100	70	100	22
controller	100	100	100	52
data	100	100	100	52

3 Test Cases

Most of the tests have to be done manually by uploading certain config files. The necessary config files for each test case are listed and can be found in `/test-case-configs`.

3.0.1 Tests for mandatory requirements

- **GTC-1** Manual test passed.

  Config files: `layout_A.yml`, `plugins_A.yml`

- **GTC-2** Manual test passed.

  Config files: `layout_A.yml`, `plugins_A.yml`

- **GTC-3** Manual test passed.

  Config files: `layout_A.yml`, `plugins_A.yml`, `general_A.yml`

  Note: The mentioned "i18n" directory can be found in `deployment/backend/build/resources` and needs not be touched as the translation file provided is already valid.

  Note: The provided general config file changes the language to german.

- **GTC-4** Manual test passed.

  Config files: `layout_B.yml`, `plugins_A.yml`

  Note: Use the editor to adjust the `from_time` and `to_time` values to the current time to test this feature (tuples containing hour and minute).

  Note: With the provided layout config only a clock will render when outside of the cafeteria time frame.

- **GTC-5** Manual test passed.

  Config files: `layout_A.yml`, `plugins_A.yml`, `calendar_data.yml`

- **GTC-6** Manual test passed.

  Config files: `layout_C.yml`, `plugins_A.yml`, `publication_data.yml`

- **GTC-7** Manual test passed.

  Config files: `layout_D.yml`, `plugins_A.yml`, `general_B.yml`

  Note: The timezone getting configured by the config files is `Europe/Berlin` and the widget is set to always display the timezone name. `Asia/Shanghai` is another valid timezone value that can be tested by using the editor and updating `default_timezone` if need be. All tz database timezone names should work

- **GTC-8** Manual test passed.

  Config files: `layout_A.yml`, `plugins_A.yml`, `calendar_data_invalid.yml`

  Note: The log entry should be “calendar fetch to [...] failed with code 404”

- **GTC-9** Manual test passed.

  Config files: `layout_A.yml`, `plugins_A.yml`, `cafeteria_data.yml`

- **GTC-10** Manual test passed.

  Note: This has been tested using a REST client and the API responds with `401 Unauthorized`.

- **GTC-11** Manual test passed.

  Note: This has been tested by just logging in with an editor account.

- **GTC-12** Manual test passed.

- **GTC-13** Manual test passed.

- **GTC-14** Manual test passed.

- **GTC-15** Manual test passed.

3.0.2 Tests for optional requirements

- **GTCO-6** Manual test passed.

  Config files: `general_C.yml`

  Note: the provided config file changes the theme to the dark theme.

- **GTCO-8** Manual test passed.

  Config files: `layout_E.yml`, `plugins_A.yml`

  Note: An album needs to be created for the image display widget to display it. We've used these images for testing: image 1 and image 2.
  The layout expects to use the album with id 1, if you create multiple albums you can alternatively change the `album_id` to the name of the album (`"Album #1"`; `#` is used for comments in yaml, make sure to wrap the name in quotes).

- **GTCO-10** Manual test passed.

- **GTCO-11** Manual test passed.

  Note: A slight change to the test case has occured.
  Instead of having to enter a name while creating the album a default name is chosen which can be edited after creating the album using the "Edit" button.

- **GTCO-12** Manual test passed.

- **GTCO-13** Manual test passed.

4 Monkey Testing

1. Open the "Dashboard" page in a browser
   The dashboard login page will be shown.

2. Enter a username and password into the text boxes
   The username will show in the text box, the password will be displayed with small dots.

3. Click "Login" button
   If the username and password are correct and you have the admin role the "Core Configuration" page will appear. If not the message "Incorrect credentials" will be displayed.

4. Click "Core Configuration" on the sidebar.

   4.1. Click the "Upload" button beneath "General", "Layout" or "Plugins".
   An upload dialog will appear.

   4.2. Chose a file.
   A success message is displayed.

   4.3. Click "Download" button beneath "General".
   The download of the general config file starts.

   4.4. Click "Edit" button for General.
   An editor opens up containing the current general config file.

   4.5. Click "Layout", "General" or "Plugins" beneath "Defaults".
   The clicked item will be ticked.

   4.6. Click "Apply Selected" button.
   The ticked items will return to the default settings. A success message is displayed.

5. Click "Plugin Configuration" on the sidebar.
   The "Plugin Configuration" page will open.

   5.1. Click one of the "Upload" buttons.
        An upload dialog will appear.

   5.2. Chose a file.
        A success message is displayed.

   5.3. Click one of the "Download" buttons.
        The download of a plugin data config file starts.

   5.4. Click one of the "Edit" buttons.
        An editor opens up containing the current config file.

6. Click "Error Logs" on the sidebar.
   A text area containing various log messages appears.

   9.5. Click the "Download log file" button in the bottom right corner.
        The download of the log file starts.

7. Click "Announcements" on the sidebar.
   The "Announcements" page will show on the right side and all existing announcements will show up.

   7.1. Enter something in the "Announcement text" text box.
        The text will appear inside of the box.

   7.2. Click "Starts" or "Expires in".
        A date picker appears.

   7.3. Click "Urgency".
        A drop down manu will show up.

   7.4. Click "Save" button.
        A success message is displayed. The new announcement appears in the table.

   7.5. Click "delete" on an Announcement in "Past active announcements"
        This Announcement will be deleted and disappears from the page.

8. Click "Calendar" on the sidebar.
   The "Calendar" page will show on the right side and all existing calendar events will show below.

8.1. Enter text in the "Name" or "Location" text box.
  The text will appear inside of the box.

8.2. Click "From" or "To".
  A date picker appears.

8.3. Click "All day".
  The checkbox gets activated.

8.4. Click "Add" button.
  A success message is displayed. The new event appears in the table.

8.5. Click "delete" on a calendar event.
  This calendar event will be deleted and disappears from the page.

9. Click "Albums Images" on the sidebar.
   The "Albums Images" page will show on the right side. And all existing Albums will show.

9.1. Click "Create new album" button.
  A new empty album will show up.

9.2. Click "Edit" on an album.
  A text box appears.

9.3. Enter a new name and click "save".
  The name of the album changes to the new name.

9.4. Enter an image URL in the "Image URL" text field of an album.
  The URL will show inside of the text box.

9.5. Click the "Add" button beneath the "Image URL" text filed.
  A thumbnail of the image will be shown.

9.6. Click the small cross in the upper right corner of an image.
  The image will be deleted from the Album and the thumbnail will disappear.

9.7. Click "delete" on an album.
  The Album will be deleted and disappears from the page.

10. Click the "Logout" button in the bottom left corner. The User will get logged out and redirected to the login page.

5 Survey

The survey was set up in two ways: If the participants had a unix based OS they could install the software and explore it on their own.
If the participants were not using a unix based OS a team member screenshared a version of the system and the participants could instruct the member to perform actions or ask questions.

After they were done the participant answered the questions below / gave their opinion regarding a certain topic.

All of the participants are students and graduates.

There were 4 participants.

5.1 Questions

How intuitive and easy to use was the system?

Answers

- The dashboard is pretty intuitive and clean. Config files require documentation to work correctly in a project.
- 7/10
- 7/10
- 5/10 A GUI instead of config files would be nicer.

How do you like the look and feel of the system and what are your thoughts on it?

Answers

- The dashboard looks very nice. Rounded corners and a drop shadow for the announcements in the frontend would have been nice but I like the colors. The clock and calendar widget look great, but the publication widget looks a bit *old*. The blurriness of the image display widget looks good.
- 7/10 The frontend has too many hard corners.
- 8/10 The light mode is pretty.
- 8/10

Did any errors occur?

Answers

- The frontend keeps flashing every time it updates, the first announcement is only displayed after a little bit of time, the time fields in the dashboard were a bit buggy (*note: all these issues have since been fixed*)
- The default layout did not work.

How configurable is the system?

Answers

- The plugin config section could see some work; ordering by backend / frontend and clearly distinguishing between them would be good. Having default config files that can be applied is pretty nice.
- 9/10 To be able to split a slot in three parts would be nice.
- 9/10
- 10/10 Good extensibility.

Is anything missing?

Answers

- The weather widget (*which was an optional requirement*) would have been nice to have. Otherwise most use cases are covered.
- The time of added Calendar events should be displayed in the dashboard.

- Show the time of added Calendar events in the dashboard.
- No.

Rate the system from 1 to 10.

Answers

- 8/10 if a few minor bugs/inconveniences can get fixed (*note: the ones in question have all been fixed*).
- 8/10
- 8/10
- 8/10

6 Bug Reports

6.1 Frontend

6.1.1 Flickering on update data

Description: Upon hitting the update endpoint and retrieving new data the frontend was flickering once for a short time while rerendering.

Steps to reproduce:

- Have the frontend open for more than a few seconds (about a minute)

Expected behaviour: No flickering on update because only relevant things should be updated; not everything.

Reason: A complex react updating scenario.

Solution: Only updating the App components state when new announcements come in (not everytime with the same data) resolves this issue mostly.

6.1.2 Announcements not being displayed instantly

Description: Announcements have a switching time. After this time the active announcement is replaced with the next one. Previously the first announcement would only be displayed after waiting the switching time once.

Steps to reproduce:

- Have at least one active announcement
- Have the frontend open

Expected behaviour: The first announcement shows up instantly

Reason: An internal counter for switching between announcements was initialized to -1 (because zero-length arrays exist and using 0 and assuming it to be valid would be foolish).

Solution: Overhauling the internal counting mechanism completely and not incorporating the array length in the counting process at all; only for the rendering.

6.1.3 CSS inheritance problem

Description: A HorizontalSplit inside of a VerticalSplit didn't render properly

Steps to reproduce:

- Use a layout config containing a hsplit inside of a vsplit

Expected behaviour: Renders properly

Reason: CSS inheritance problem were the styling ment for each child of the VerticalSplit superceeded the styling ment for each child of the HorizontalSplit

Solution: Only apply styling for direct descendants.

6.1.4 Layout component errors not being displayed

Description: Previously errors in the layout'ing process weren't displayed.

Steps to reproduce:

- Upload a plugins config which doesn't contain all the widgets that the layout config uses

Expected behaviour: The error is displayed to the user in some kind of way.

Reason: Until recently the errors weren't getting detected at all so the possibility of displaying them weren't there.

Solution: Added ways of displaying the errors.

6.1.5 No checks for layout containing plugins not mentioned in plugins.yml

Description: When the layout uses plugins which aren't mentioned in the plugins.yml file no error was previously generated

Steps to reproduce:

- Upload a plugins config which doesn't contain all the widgets that the layout config uses

Expected behaviour: An error is generated

Reason: This wasn't implemented prior.

Solution: Visit each node of the layout tree and compile a list of all referenced plugins. This list is then later checked against the expected list of plugins and if it isn't a subset an error is generated.

6.1.6 CarouselSlot bug

Description: The CarouselSlot couldn't handle having zero slots passed to it (either through initially setting it to zero or by updating it to it).

Steps to reproduce: (these are the almost the exact steps the test "CarouselSlot can handle removal of slot" also does)

- Render CarouselSlot with multiple widgets (for example 3)
- Update props to make it zero widgets

Expected behaviour: The CarouselSlot doesn't render anything.

Note that if you test this with the normal frontend this shouldn't be able to even happen as the layout system does a cleaning of the layout tree prior to rendering anything. This fix just fixes some edge cases which may or may not be able to arise at all.

Reason: Code wasn't ment to be used with no slots.

Solution: Clean up the code of the CarouselSlot a lot which then made it easy to implement an extra check for this scenario.

6.1.7 ImageDisplay z-index

Description: When using the ImageDisplayWidget alongside announcements it could happen (depends on image size, slot size and screen aspect ratio) that the inner (non-blurred background-) image would be displayed ontop of announcements.

Steps to reproduce:

- configure the `Plugins` in `Core configuration` to set the image display plugin active.
- configure the `Layout` in `Core configuration` and place the image display plugin in the bottom part.
- If there doesn't exist active announcement, go to the `Announcementpage` and add a new announcement.
- If there is no album source in the image display plugin, go to the `image page` and add a new album (not empty) into it.
- if done just right, part of the announcement is overlapped by the image.

Expected behavior: The announcement is displayed ontop of the image.

Reason: z-index property of inner image.

Solution: Remove unneeded z-index css property of inner image.

6.2 Dashboard

6.2.1 Invalid defaults file

Description: Using the defaults file for the layout would cause a problem because it didn't have the right structure.

Steps to reproduce:

- Open the `CoreConfigurationPage` and try to apply the defaults file for layout
- See it fail because the file has an invalid structure

Expected behaviour: The file works.

Reason: The file had an invalid structure.

Solution: Fix the structure.

6.2.2 TimeInput bug

Description: All TimeInput's displayed would somehow be somewhat sync'ed. Every update to a TimeInput would only trigger the onChange callback on the last rendered TimeInput component instead of actually updated components onChange prop.

Steps to reproduce:

- Go to the `CalendarPage` and try changing the start time.
- The end time will update instead of the start time.

Expected behaviour: The end time remains untouched and the start time updates.

Reason: A complex react issue having to do with functional components, their state, different execution stages of the functional component (mainly normal code in the body of the function, code inside of useEffect, code inside of the JSX being returned).

Solution: Rework the functional component to a classical class component and refactor the code a bit outright.

6.2.3 Incorrect headers

Description: The dashboard would send an `Authentication`-header instead of a `Authorization`-header as expected by the backend.

Steps to reproduce:

- Trigger any api call that requires to be authenticated (for example uploading a config file).
- Watch the request fail with a 401.

Expected behaviour: Request succeeding.

Reason: Wrong header was being sent.

Solution: Change the header to `Authorization`.

6.2.4 Table component

Description: The alignment of the data doesn't meet the `dataAlignment` set in the props.

Expected behaviour: The alignment of the data meets the `dataAlignment` set in the props.

Reason: Instead of `dataAlignment`, the `headingAlignment` is set to the `get_alignment()` function in style of the data.

Solution: Change the `headingAlignment` to `dataAlignment`.

6.2.5 ErrorLogPage can not be visited.

Description: A compile error appears when trying to enter the `ErrorLogpage`.

Steps to reproduce:

- Click the `ErrorLogpage` in dashboard and an error will occur.

Expected behavior: Successfully open the `ErrorLogpage` and see an Errorlog table as well as a `download_errorlog` button.

Reason: Sometimes the error log returned by `parsed_line` function is undefined. So it may cause `undefined` problem when we try to render the `ErrorLogpage`.(we need to call the `urgency`,`timestamp` and `message` of an error log.)

Solution: Added an extra `if`-statement for this specific situation. It will simply render an empty `errorlog` table instead.

6.3 Backend

6.3.1 Unable to update refresh time

Description: It is not possible to update the refresh period of jobs e.g. fetching of Publications.

Steps to Reproduce: Change the refresh time in e.g. the publication data config file.

Reason: The refresh time is special and can't be updated like other config values as it doesn't change anything "internal" to the plugin but "external" (the job scheduler needs to update the delay value) and therefore needs to be handled somewhat differently. This scenario only happens with this config value.

Solution:
Added extra `reschedule` method to `JobSchedulerService`.
Added `DataConfig` abstract class. Using the functionality provided by the new class a service will now be notified when the refresh time changes and can use `reschedule` to change the refresh period of all jobs.

6.3.2 Unspecific error codes

Description: Some error codes are not very specific.

Steps to Reproduce: Delete an announcement with an ID not used by any announcement.

Behavior: A 500 internal server error is returned.

Expected behavior: A 404 not found error is returned.

Reason: Often a default error handler was used instead of specially handling the error.

Solution: Special error handlers were implemented.

6.3.3 Invalid issuer encoded in JWT

Description: When the backend is running inside a docker container authorization always fails.

Steps to Reproduce: Use a valid token and try to access a secured endpoint e.g. `announcements` to create an announcement.

Behavior: A 401 unauthorized error is returned.

Expected behavior: The status code is 200 and the new announcement is returned.

Reason: The backend expects the issuer of the token to be *keycloak*, but the actual issuer is *localhost*.

Solution: Reconfiguration of docker containers.

6.3.4 LocalRepository.findByID can crash

Description: In `LocalRepository` the method `findByID` crashes when a value is not found.

Behavior: Crash or internal server error.

Expected behavior: An empty optional is returned.

Reason: The method `Optional.of` instead of `Optional.ofNullable` was used.

Solution: `findByID` is now using `Optional.of`.

6.3.5 YAML parsing library inconsistencies

Description: The YAML parsing library sometimes unexpectedly does not return a map e.g. on invalid YML.

Behavior: Crash or internal server error.

Expected behavior: A map with the mapping __config_file to file content is returned.

Reason: The library is not well documented.

Solution: An extra check was added to ensure the cast to a map is successful.

6.3.6 Calendar events race condition

Description: If two or more events, start at the same time only one is returned.

Steps to Reproduce:

- Create multiple calendar events.
- Use the `calendars/internal` endpoint to list the events.

Behavior: Even if multiple events exist only one is returned.

Expected behavior: All events of the calednar are returned.

Reason:
To sort the events a TreeSet was used. If two events start at the same time they are considered equal, thus events with the same starting time are discarded.

Solution: The compare method now also checks the ID of the event.

7 Log List

7.1 Backend

- Add `parseWeekDay, parseIntList, getAllDates, expandDatesOn, getUpperUnit, isDateValid and handleNegativeValues` methods to `ExternalCalendarService` class.
  Reason: The methods are used for implementing the recurrence of calendar events and make the code easier to understand, read and maintain.

  The fetch methods in `PublicationService` `ExternalCalendarService` and `CafeteriaService` were split up into two kinds of methods, one for parsing data and one for retrieving data.
  Reason: This makes the code way easier to test and extend the code.

- Added new abstract class `DataConfig`.
  Reason: Using the functionality provided by the new class a service will now be notified when the refresh time changes.

- Add `reschedule` method to `JobScheduler`.
  Reason: Necessary to update the refresh period (fixes the bug).

- Removed `updateImage` method from `ImageDisplayService`.
  Reason: The method is not needed because `saveImage` can be used instead.

- Some more methods now throw `EntityNotFoundException`.
  Reason: This is needed to generate more specific error status codes.

- New field `verbose` in class `Backend`.
  Reason: The field can be set via the `-verbose` flag and causes status messages to be logged.

- Added `deleteAllDishe` method in class `Line`.
  Reason: This is used to make sure not to much disc space is used.

7.2 Frontend

- App now passes `expected_plugins` to the Layout component.
  Reason: The Layout component checks wether plugins are referenced in the layout which aren't listed in the plugins.yml file. For this to work the layout needs the list of plugins from the plugins.yml file. Since the App component loads this file the expected plugins were just added as a prop to the Layout component.

- Contexts are being used less in favor of just manually importing the stores in the plugins.
  Reason: Not using contexts comes with a few advantages when it comes to rendering and mobx stores as making something observable works better when it isn't wrapped in a higher order function.

- Layout has new utility functions `visit_widgets`, `get_all_widgets` and `clean_tree`
  Reason: These functions are used for the improved "Time configurable-widget display" and checking if the layout mentions plugins which aren't in the plugins.yml file.

- PluginManager (src/plugins/index.ts) now has `load_plugin_from_cache` in addition to `load_plugin`.
  Reason: The time when plugins are loaded has changed a bit, the layout now does this upfront instead of relying on every WidgetSlot to initiate this himself. `load_plugin` used the cache if the plugin was already loaded but tried making a network request if it wasn't. Therefore it returned a Promise which made testing a bit harder as testing is way easier if you don't have to keep the event loop in mind and can just rely on things being done at a specific point. Since the layout prefetches all plugins they will definitely be in the cache at the time the WidgetSlot tries to render them. Henceforth a way to load things only from the cache without having a Promise returned was created.

7.3 Dashboard

- The Albums & Images page now exists.
  Reason: It wasn't implemented before.