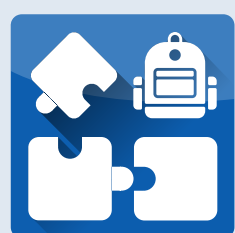


Dominik George, Tom Teichler, Jonathan Weth, the AleksIS® Team and contributors

The AleksIS® Handbook

Release 2025.1.1

Project Information · Installation and Administration
End-User Handbook · Developer Manual



ALEKSIS

THE FREE SCHOOL INFORMATION SYSTEM

Dominik George, Tom Teichler, Jonathan Weth, the AleksIS® Team and contributors

The AleksIS® Handbook

Release 2025.1.1

Project Information · Installation and Administration

End-User Handbook · Developer Manual



This page intentionally left blank.

CONTENTS:

1	Preamble	1
1.1	Introduction to AleksIS	1
1.1.1	Welcome	1
1.1.2	On School Information Systems	1
1.1.3	AleksIS, the Free School Information System	2
1.1.4	The wider digital educational ecosystem	2
1.2	The official AleksIS standard distribution	3
1.2.1	Releases and support	3
1.2.2	Apps and versions	3
1.2.3	Installation of the standard distribution	3
1.3	Highlights and notable changes in this release	3
1.3.1	New apps and features	4
1.3.2	Notable changes in existing features for users	4
1.3.3	How to upgrade	5
2	End-user handbook	7
2.1	Using basic features of the AleksIS Core	7
2.1.1	Register a new account	7
2.1.2	Managing your personal account	8
2.1.3	Selecting an active school term	12
2.1.4	Dashboard	12
2.1.5	PWA (progressive web application)	14
2.1.6	AleksIS' calendar system	17
2.1.7	Personal events	17
2.1.8	WebDAV access	18
2.2	Timetables and substitution plans with Chronos	19
2.2.1	Finding information about the timetable	19
2.2.2	Substitution plan	22
2.2.3	Notifications about current changes	23
2.2.4	Managing substitutions	23
2.3	Digital class register	25
2.3.1	Concept of Alsijil and overview about functionality	25
2.3.2	Basic data concepts	25
2.3.3	Overview about lessons (Coursebook)	26
2.3.4	Overview about absences	29
2.3.5	Statistics	30
2.3.6	(Archive) printout	31
2.4	Using seating plans	31

2.4.1	Managing seating plans	31
2.4.2	Integration with class register	33
2.5	Time-based documents with Resint	34
2.5.1	Publishing posters	34
2.6	Get support via Hjelp	35
2.6.1	Reading Frequently Asked Questions	35
2.6.2	Reporting issues	36
2.6.3	Giving feedback for AleksIS	37
3	Administration handbook	39
3.1	Setup of the AleksIS instance and the AleksIS core	39
3.1.1	Concepts of the AleksIS core	39
3.1.2	Install AleksIS	42
3.1.3	Configuration files and format	48
3.1.4	Configuration options	48
3.1.5	Storage	50
3.1.6	Mail	52
3.1.7	Authenticating against LDAP	52
3.1.8	Registration and user invitations	53
3.1.9	Social accounts	54
3.1.10	Monitoring and health checks	56
3.1.11	Background tasks	57
3.1.12	Data checks	58
3.1.13	Providing important information to users using the dashboard	59
3.2	Managing timetable data and substitution plans in Chronos	61
3.2.1	Managing timetable and substitution data	61
3.2.2	Setup notifications about current changes	62
3.2.3	System-wide settings for timetable and substitution display	62
3.3	Setting up a digital class register with AleksIS-App-Alsijil	63
3.3.1	Defining base data	63
3.3.2	System-wide settings for the digital class register	64
3.4	Seating plan administration	64
3.4.1	Permissions	64
3.5	Providing time-based documents with Resint	65
3.5.1	Uploadable posters for time-based documents	65
3.5.2	Configuring live documents	66
3.6	Importing timetables and substitutions from Untis	67
3.6.1	Untis data and their relation to AleksIS	67
3.6.2	Setting up the Untis integration	68
3.7	Importing data from CSV-like sources	70
3.7.1	Base concept of the generic importer	70
3.7.2	Configuring the import	71
3.7.3	Exporting data from various systems	71
3.7.4	Importing data	72
3.7.5	Importing data from Untis	72
3.8	Show data from external sources on dashboard	74
3.8.1	Introduction	74
3.8.2	RSS feed widget	74
3.8.3	iCalendar feed widget	76
3.9	Advanced LDAP integration	77
3.9.1	Interfacing with an LDAP directory beyond authentication	77

3.9.2	Configuring LDAP synchronisation	77
3.9.3	Managing user accounts in LDAP	79
3.10	Setting up Matrix synchronization	79
3.10.1	Matrix and Element	79
3.10.2	Concepts of rooms and spaces	80
3.10.3	Configuring Matrix connection	80
3.10.4	Synchronizing with Matrix	81
3.11	Support end users and get feedback with Hjelp	82
3.11.1	Managing FAQs	82
3.11.2	Managing issue report form	83
3.11.3	Settings and preferences	85
4	Developer handbook	87
4.1	Development of AlekSIS apps using the AlekSIS core	87
4.1.1	Setting up the development environment	87
4.1.2	Installing apps into development environment	89
4.1.3	Running tests and reports	90
4.1.4	Materialize templates	91
4.1.5	Vue-based frontend	94
4.1.6	Extensible models	94
4.1.7	Merging of app settings	94
4.1.8	Registering dashboard widgets	95
4.2	Providing auto-generated documents from apps	96
4.2.1	Providing live document types from other apps	96
5	Appendix	97
5.1	About Free Software and Open Source	97
5.1.1	The four freedoms	97
5.1.2	Advantages for schools	98
5.2	Legal Information about the AlekSIS® Software	98
5.2.1	Copyright Note and Licenses	98
5.3	AlekSIS® trademark	99
5.4	EUROPEAN UNION PUBLIC LICENCE v. 1.2	99
5.4.1	1. Definitions	99
5.4.2	2. Scope of the rights granted by the Licence	100
5.4.3	3. Communication of the Source Code	101
5.4.4	4. Limitations on copyright	101
5.4.5	5. Obligations of the Licensee	101
5.4.6	6. Chain of Authorship	101
5.4.7	7. Disclaimer of Warranty	102
5.4.8	8. Disclaimer of Liability	102
5.4.9	9. Additional agreements	102
5.4.10	10. Acceptance of the Licence	102
5.4.11	11. Information to the public	103
5.4.12	12. Termination of the Licence	103
5.4.13	13. Miscellaneous	103
5.4.14	14. Jurisdiction	103
5.4.15	15. Applicable Law	103
5.4.16	Appendix	104

Bibliography	105
---------------------	------------

PREAMBLE

1.1 Introduction to AleksIS

1.1.1 Welcome

Welcome to AleksIS, the Free School Information System! This handbook is a comprehensive guide on how to install, administer, and use AleksIS and all included official apps. Its target audience are system administrators seeking to install and maintain an AleksIS installation, people supporting the use of AleksIS in daily processes in educational organisations, and end users who work with the software on a daily basis.



1.1.2 On School Information Systems

School Information Systems, or for short, SIS, are software suites that help educational organisations with managing personal information about all members (e.g. students, teachers, parents and guardians) as well as organisational data like plans, pedagogic track records, and much more.

Generally, the goal of using School Information Systems is to help members of the school or other educational organisation to organise their work in learning and teaching, and to simplify digital processes, especially those that involve analytical evaluation, statistics, or information that is updated very frequently and should be made available to its recipients in a timely manner.

Ideally, School Information Systems also promote transparency between teachers and students and improve sustainability by minimising the need for paperwork.

1.1.3 AleksIS, the Free School Information System

School Information Systems have been on the rise ever since schools began digitising their management processes. Traditionally, the vendors of school management tools (like for timetabling, personal data management of teachers and students, etc.) also deliver web applications that are tailored towards making the respective data available to those who need it.

As most of these solutions are proprietary products, they tend to be limited to be used together with other software of their vendors, and do not integrate nicely with other software and with processes and data sets not in the focus of the developers.

AleksIS, in contrast, is a generic School Information System that aims at being universally usable for all processes and data sets in any educational organisation, and at interacting closely with other software components employed in management and lessons.

The development and product team has set the following goals for AleksIS, in their dedication to support learning and teaching in the digital world:

- Create a high-quality software product, adhering to high standards regarding data protection, data safety, security, and stability
- Comply with as many national and regional data protection and school laws as possible (cf. [Legal Information about the AleksIS® Software](#) (page 98))
- Provide an extensible framework, allowing schools or their supporting IT companies to create their own apps (cf. [Developer handbook](#) (page 87))
- Help coding clubs and classes to understand the development as deeply as necessary to use AleksIS for learning and creating their own apps
- Keep AleksIS free, open, and transparent (cf. [About Free Software and Open Source](#) (page 97))

1.1.4 The wider digital educational ecosystem

AleksIS is a School Information System, and as such, is intended to be a superior tool for data management, and for digitising management processes in education.

As laid out in the Basics of the Unix Philosophy, good software should “do one thing and do it well” [[UnixPhil](#)]. Therefore, AleksIS will not provide features that are covered by other good and free software. The following examples list features that are out of scope for AleksIS:

- **Messenger features:** There are good free software communication platforms, e.g. [Element](#)¹
- **Learning Management:** Managing online courses, tests, etc. is covered by systems like [Moodle](#)²
- **File sharing and groupware:** The [Nextcloud](#)³ private cloud is a good solution for calendars, file sharing, and other groupware needs

AleksIS strives to provide integrations with popular other platforms, and to ease the process of developing such integrations.

¹ <https://element.io/>

² <https://moodle.org/>

³ <https://nextcloud.com/>

1.2 The official AleksIS standard distribution

AleksIS consists of a core, which provides all general functionality and management of basic data, and additional apps that can be installed in a modular fassion.

1.2.1 Releases and support

The default distribution is a meta-package that defines which apps are officially supported, and what the development team canonically calls *AleksIS®*. It is released every 6 months, aligned with the beginning school half-year in summer and winter.

In the official distribution, the core and apps are semantically versioned and limited to their minor version. That means that, if the standard distribution is installed through the meta-package, all apps are kept up to date with bugfixes, but do not receive feature updates or even breaking changes.

The official distribution is supported for one year (in other words, the most recent two distribution versions are supported). The supported versions receive security updates and important bugfixes.

1.2.2 Apps and versions

App name	Description	Version
Core	Core functionality and framework	4.0.4
Cursus	Subjects and courses	1.0.0
Lesrooster	Timetable management	1.0.0
Chronos	Timetables and substitutions	4.0.0
Kolego	Absences and staff management	1.0.0
Alsijil	Class register	4.0.0
Untis	Import from Untis	3.0
Hjelp	Support, feedback and FAQ	4.0.0
DashboardFeeds	Dynamic external feeds as dashboard widgets	4.0.0
CSVImport	Import generic CSV files	3.0
LDAP	LDAP synchronisation	4.0.0
Resint	Time-based/live documents	4.0.0
Matrix	Integration with Matrix/Element	3.0.0

1.2.3 Installation of the standard distribution

The standard distribution is primarily available as the `aleksis` package on PyPI. All installation documentation applies, and to install the standard distribution, the package `aleksis` can be used in all places where `aleksis-core` is installed for custom installations.

1.3 Highlights and notable changes in this release

This release of the AleksIS® distributions features the following notable changes for administrators and users, compared to the previous official release.

1.3.1 New apps and features

Global calendar system

Everything that fits in a calendar is accessible in one place. In a typical school day, not only the regular timetable is relevant, but also holidays and personal events like meetings and birthdays. Therefore, AleksIS features a global calendar view, in which single calendars (e.g. Lessons, Personal Events and Holidays) can be activated or deactivated individually to get the desired overview.

To view availabilities easily, there is a dedicated free/busy overview, aggregating availabilities of single persons or groups.

Contacts and calendar synchronisation

AleksIS now supports the open protocols CardDAV and CalDAV to synchronize contacts, i.e. persons and organisations, and calendar feeds to external applications. Currently, this feature is read-only, so changes can only be made in AleksIS.

Global school term select

Most of the time, one will work in the current school term. But in specific situations it can be useful to switch into the past, e.g. for viewing an old class register, or into the future to plan the timetable for next school year.

Timetable creation

Our big new feature in this release is timetable creation. While previously it was only possible to import timetables from external sources, they can now be managed in AleksIS itself, so that no additional software is required anymore.

Modern class register and course book

The class register has been re-worked from the ground, integrates with the calendar and absences system to use and provide information globally and has a flexible interface that allows filtering by courses and groups, as well as only showing incomplete documentations. Participants can be marked as present, late or absent conveniently.

Absences and substitution planning

Teacher's absences entered in AleksIS are evaluated to generate a list of affected lessons, to simplify the process of substitution planning.

1.3.2 Notable changes in existing features for users

- Menu entries for school terms and announcements have been moved to the new category data management, which also includes rooms and holidays.
- Specific support for Yubikey as a second factor for authentication has been removed. The open standard Webauthn is still supported.
- iCal widgets were removed.

1.3.3 How to upgrade

To upgrade to the current release, simply install the new version as laid out in the install instructions. These instructions are supposed to be idempotent and can simply be repeated.

The setup instructions are detailed at [Install AleksIS](#) (page 42).

The functionality of the new apps Cursus, Kolego and Lesrooster was mainly covered by AleksIS-App-Chronos in the past. Lesrooster contains a migration path for migrating 3.x versions of AleksIS-App-Chronos and AleksIS-App-Alsijil. If you want to migrate from these versions, please be sure that the apps are installed. Then run migrations as usual. Please be aware that depending on the amount of data the migration might take a while. On really large installations, several hours are to be expected. In case you run into memory issues, please contact us for support.

Required manual actions after upgrade

Administrators are required to do the following manual steps on or after upgrade:

- Clearing the browser cache might be necessary in some situations after upgrading.
- For the calendar system, AleksIS now needs an extension for the PostgreSQL database. Please check the installation instructions on how to setup the `pg_r rule` extension for PostgreSQL.
- AleksIS now uses Valkey as a drop-in replacement for Redis. Please update your configuration files accordingly (see installation instructions for further instructions).
- To make setting names consistent, the setting `auth.login.registration.unique_email` was renamed to `auth.registration.unique_email`. Adjust your config if necessary.
- The “additional fields” feature has been removed because it had design issues that practically made it unusable in all scenarios. No migration path away from the feature is defined. If you have been using additional group fields and need a replacement, please contact the development team.

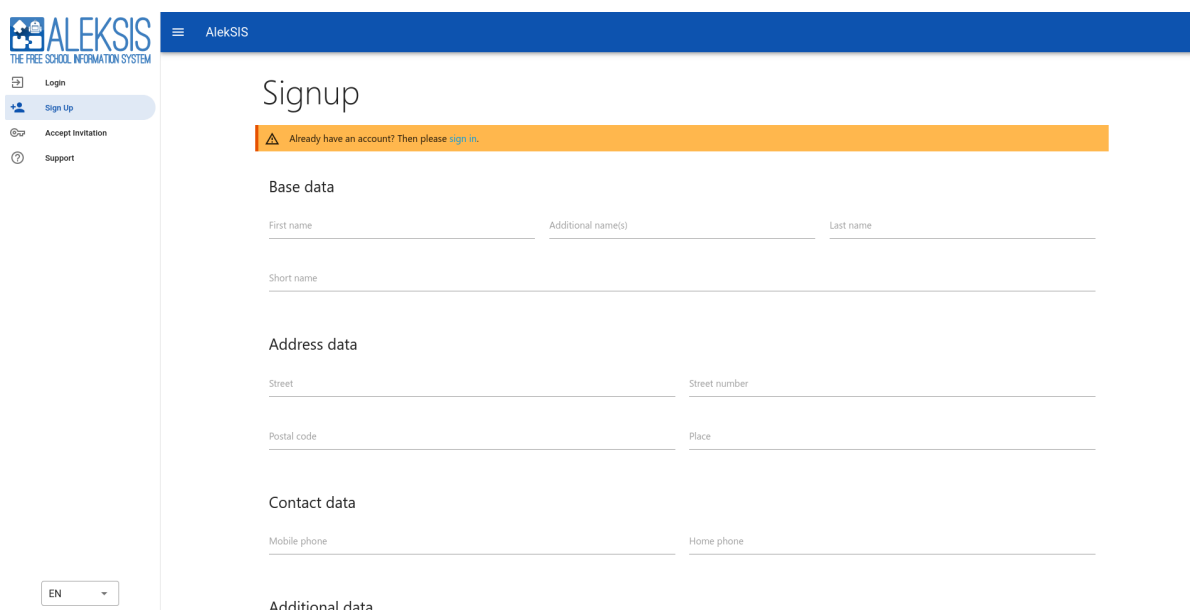
END-USER HANDBOOK

2.1 Using basic features of the AlekSIS Core

2.1.1 Register a new account

Public registration

If public registration is enabled on the AlekSIS instance, you can click the Signup button located in the navigation sidebar and register for an AlekSIS account.

The screenshot shows the AlekSIS web interface. On the left is a navigation sidebar with the AlekSIS logo and links for Login, Sign Up, Accept invitation, and Support. The main content area is titled 'Signup' and features a blue header bar with the AlekSIS logo and a hamburger menu. Below the header, there is a yellow warning banner that reads 'Already have an account? Then please sign in.' The form is divided into three sections: 'Base data' with fields for First name, Additional name(s), Last name, and Short name; 'Address data' with fields for Street, Street number, Postal code, and Place; and 'Contact data' with fields for Mobile phone and Home phone. At the bottom of the form is an 'Additional data' section. A language dropdown menu is visible in the bottom left corner, currently set to 'EN'.

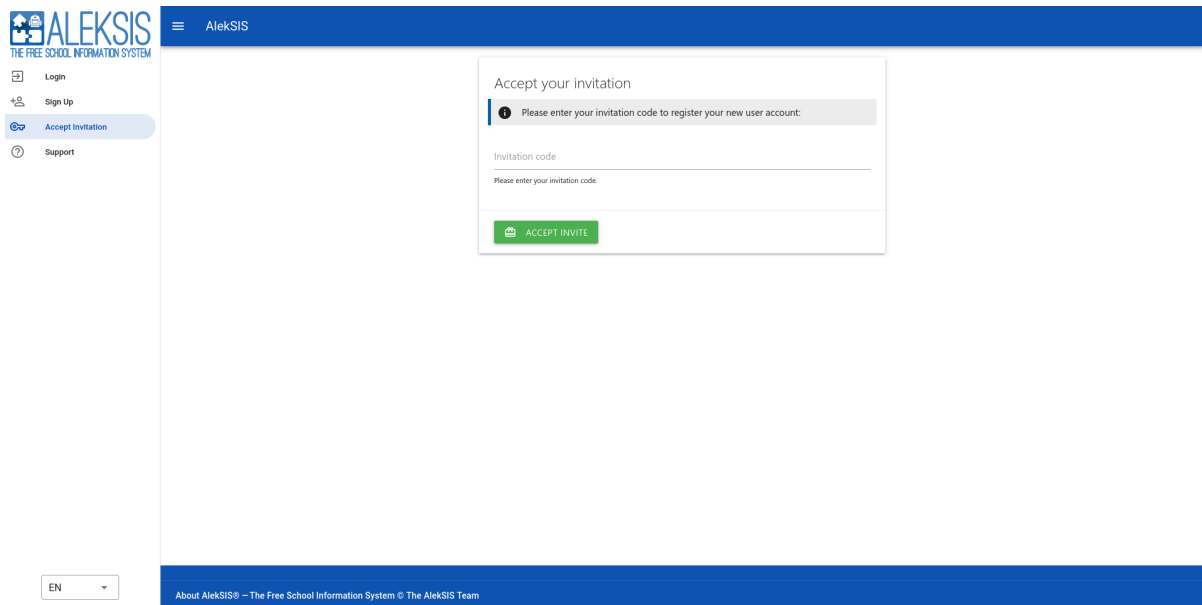
If enabled, you have to verify your email address after signup. To do so, click on the link you received on the email address you entered in the signup form.

Note

Normally, AlekSIS does not allow public registration, and all accounts are centrally managed by administrators. Public registration is an optional feature that has to be enabled by administrators after careful consideration.

Using an invitation code

If you have an invitation code, click **Accept invitation** in the sidebar and enter it. You will be redirected to the signup form.



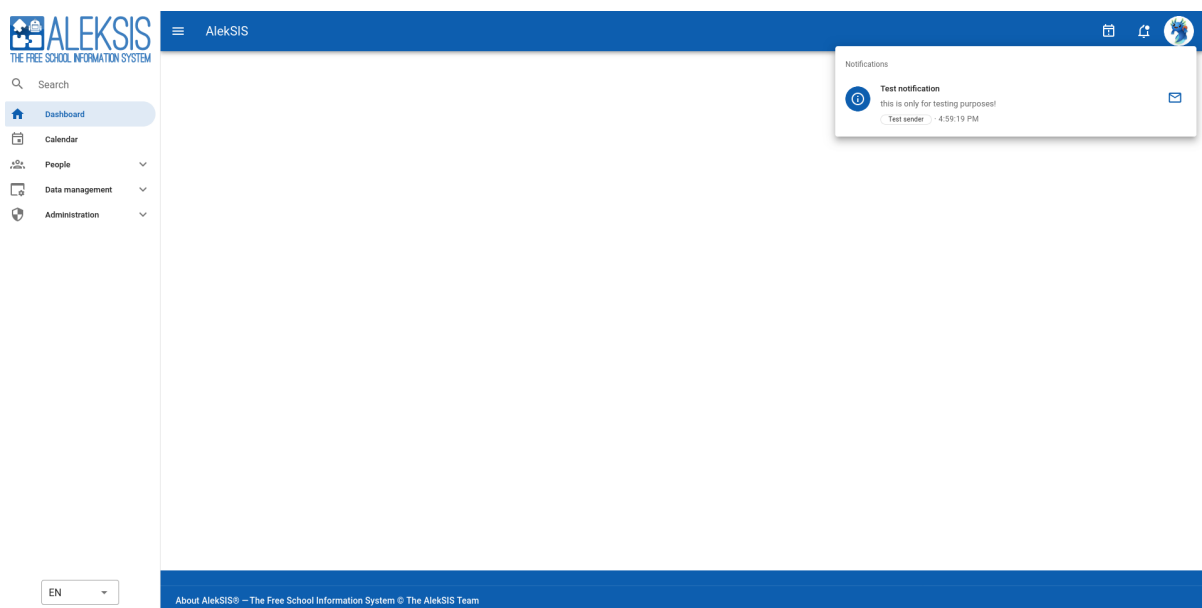
The screenshot shows the AleksIS web interface. On the left is a sidebar with the AleksIS logo and navigation links: Login, Sign Up, **Accept Invitation** (highlighted), and Support. The main content area is titled 'Accept your invitation' and contains a message: 'Please enter your invitation code to register your new user account:'. Below this is a text input field labeled 'Invitation code' with a placeholder 'Please enter your invitation code.' and a green 'ACCEPT INVITE' button. The footer includes a language selector set to 'EN' and a copyright notice: 'About AleksIS® – The Free School Information System © The AleksIS Team'.

If you've received an invitation link (e.g. via email), clicking the link will redirect you to the signup form automatically.

2.1.2 Managing your personal account

Each logged in user has several options to provided through the AleksIS core. Which of these items are display depends on whether the user has a person and what your system administrator has configured. All of the functionality listed below (except of the notification menu) can be accessed via the account menu that is shown when clicking the own avatar on the right of the app bar.

Notifications

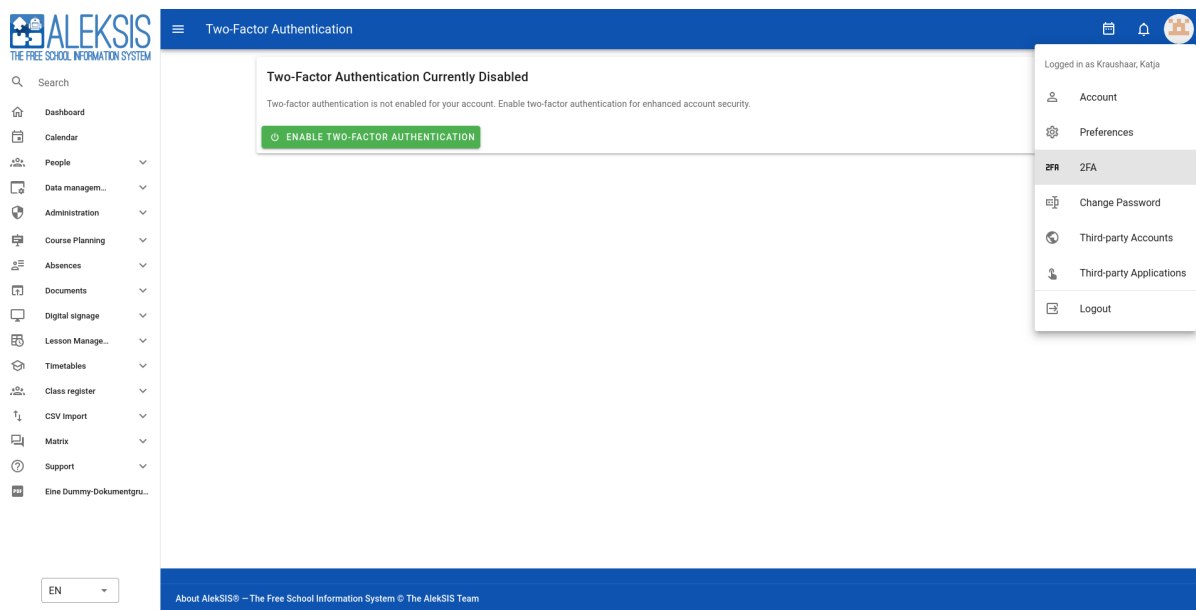


The screenshot shows the AleksIS dashboard. The sidebar on the left includes the AleksIS logo, a search bar, and a menu with 'Dashboard' (highlighted), 'Calendar', 'People', 'Data management', and 'Administration'. The main content area is mostly blank. A notification pop-up is visible in the top right corner, titled 'Notifications' and containing a 'Test notification' message: 'this is only for testing purposes!' with a 'Test sender' and a timestamp '4:59:19 PM'. The footer is identical to the previous screenshot, showing the language selector and copyright notice.

The AleksIS core has a built-in notification system which can be used by apps to send urgent information to specific persons (e. g. timetable changes). Notifications are shown in the notifications menu reachable by the bell icon in the app bar. In addition to that, notifications can be sent to users through several communication channels. These channels can be switched on or off in your personal preferences (cf. [Personal preferences](#) (page 11)).

Notifications can be marked as read using the mail button on the item's right side.

Setup two-factor authentication



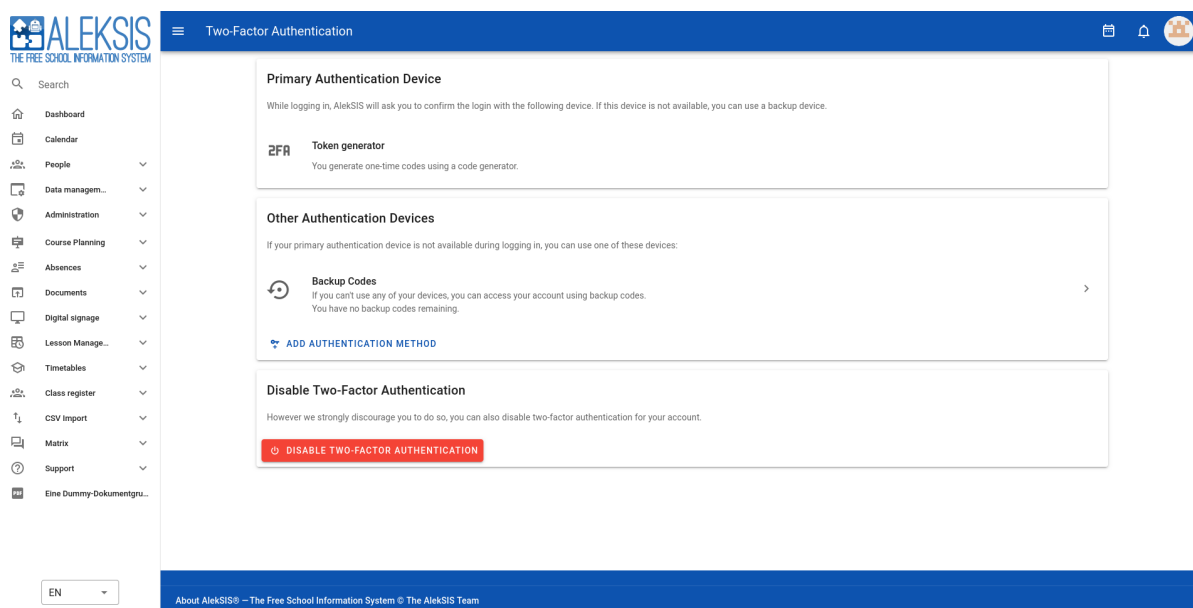
AleksIS provides two factor authentication using hardware tokens such as yubikeys which can generate OTPs or OTP application. Additionally, all devices are supported that make use of FIDO U2F.

To configure the second factor, visit **Account menu → 2FA** and follow the instructions.

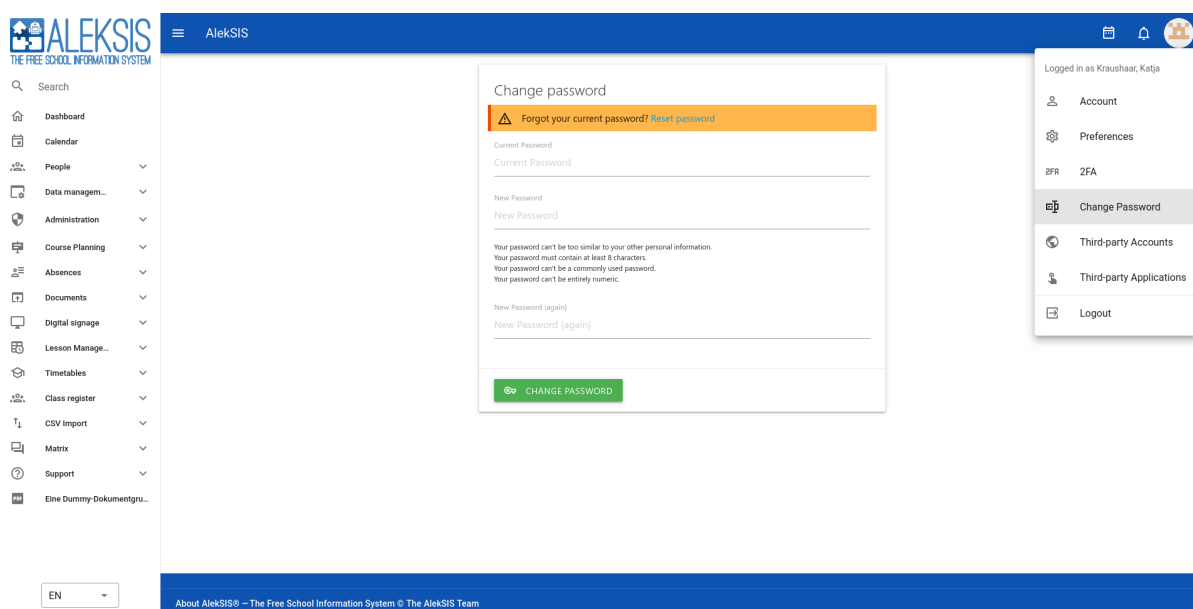
Please keep the backup codes somewhere safe so you do not lose access to your account. If you are unable to login with two factor authentication, please contact your site administrator.

If you forget to save your backup codes, but you are still logged in, visit **Account menu → 2FA**, and press the arrow to the right of the **Backup Codes** item in order to view existing backup codes or to generate new ones.

To disable two factor authentication, login to your account and navigate to **Account menu → 2FA**, then press the big red button to disable 2fa.

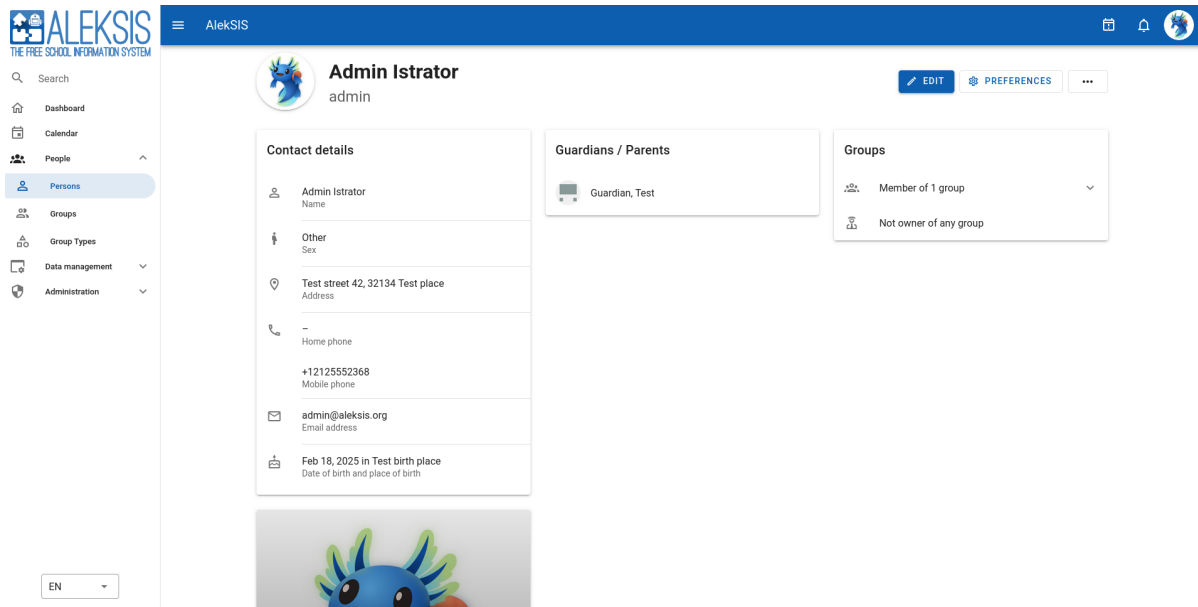


Change password



If your system administrator has activated this function, you can change your password via Account menu → Change password. If you forgot your password, there is a link Password forgotten? on this page which helps with resetting your password. The system then will send you a password reset link via email.

Me page



Reachable under Account menu → Account, this page shows the personal information saved about you in the system. If activated, you can upload a picture of yourself or edit some information using the Edit button.

Apps can extend the information shown on this page by adding widgets displaying other personal data, such as coursebook statistics or absences.

Personal preferences

You can configure some behavior using the preferences under Account menu → Preferences. By default, the Core only provides some preferences, but apps can extend this list. You can find further information about such preferences in the chapter of the respective apps.

• Notifications

- **Name format for addressing:** Here you can select how AleksIS should address you.
- **Channels to use for notifications:** This channel is used to sent notifications to you (cf. [Notifications](#) (page 8)).

• Calendar

- **First day that appears in the calendar:** Here you can select first weekday that is shown in the calendar frontend.
- **Activated calendars:** These calendars are shown in the calendar select list in the calendar frontend.

Third-party accounts

If you logged in using a third-party account (e. g. a Google or Microsoft account), you can manage the connections to these accounts on the page Account menu → Third-party accounts.

The feature to use third-party accounts needs to be enabled by an administrator, as described in [Social accounts](#) (page 54).

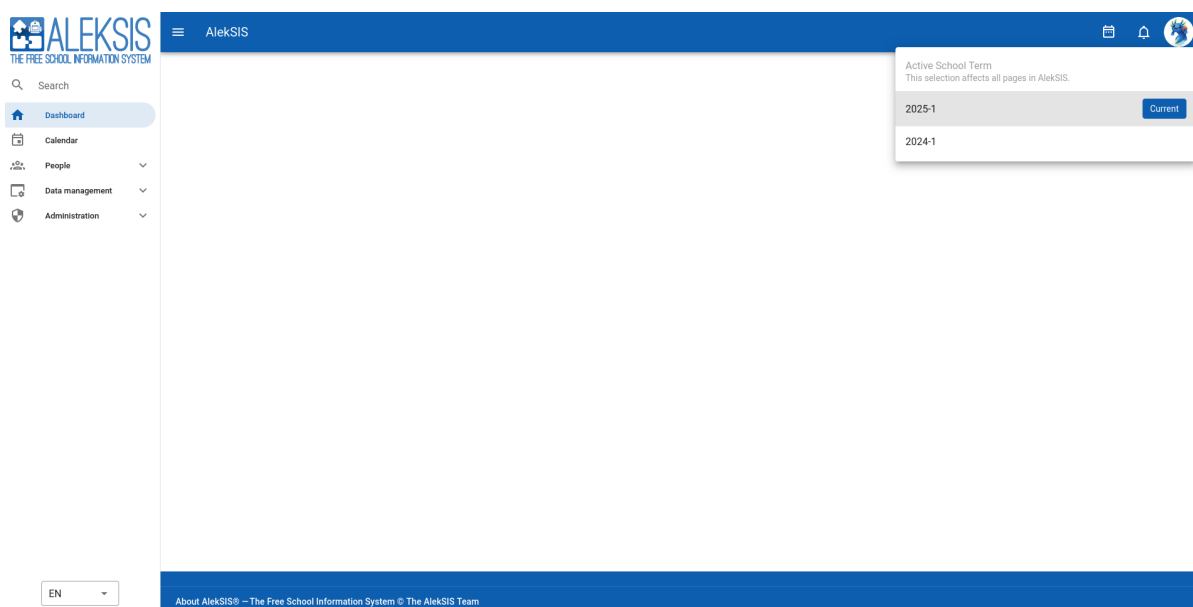
Authorized third-party applications

On the page Account menu → Third-party applications you can see all external applications you authorized to retrieve data about you from AlekSIS. That can be services provided by your local institution like a chat platform, for example.

For each third-party application, you can see the personal information it has access to. Additionally, you may revoke its access.

2.1.3 Selecting an active school term

Some information that is shown in AlekSIS are optionally bound to a specific time period called *school term*. This includes groups, for example. Other apps may add more categories of data that are also bound to school terms, such as lessons or coursebook entries. When viewing or editing these kinds of information, you need to have an active school term selected. By default, the *current* school term is selected, but in case you want to access archived data, selecting a past school term is possible. Selecting school terms can be done in the menu accessible by clicking the calendar icon in the app bar.



2.1.4 Dashboard

The first thing you will see after the login is the dashboard. Depending on what your system administrator configured, you will be able to see information from different apps at one glance.

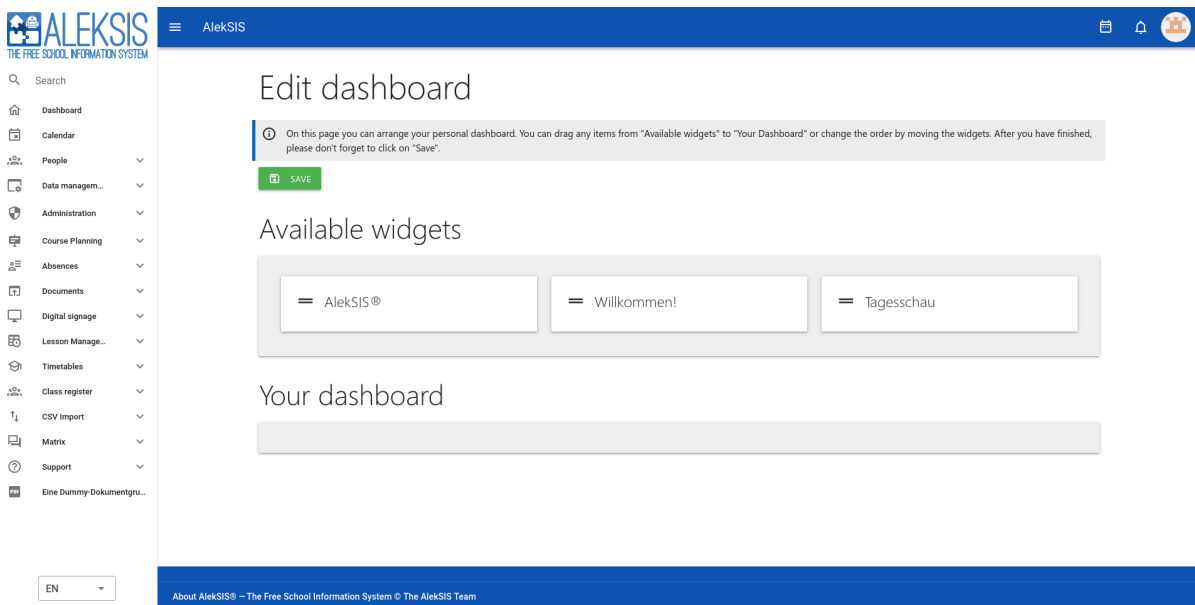


Dashboard widgets

The dashboard consists of different parts, the so-called *dashboard widgets*. They are configured by the system administrator and can be freely arranged on the dashboard (cf. [Customising the dashboard](#) (page 13)).

Customising the dashboard

There are several options for customising your personal dashboard. By default, you will see a layout provided by your system administrator. Using the button *Edit dashboard* on the top right corner of the dashboard, you can change the selection and position of the widgets.



On the edit page, you will see a list of all available widgets and your current dashboard. If the section *Your dashboard* is empty, the default dashboard will be shown. To make an own layout, you can drag widgets from the *Available widgets* to *Your dashboard*. Within *Your dashboard* you also can arrange the widgets by dragging them. To remove widgets from the dashboard, you just have to drag them back to *Available widgets*.

In addition to editing the dashboard, you can also change some preferences referring to the dashboard. This is done under the menu item *Account* → *Preferences* → *General*:

- **Automatically update the dashboard and its widgets:** If enabled by you and the system administrator, the dashboard will be updated automatically every 15 seconds.

2.1.5 PWA (progressive web application)

What is a progressive web application?

A PWA is an application developed with common web technologies and delivered in form of a website, but which offers some features a traditional website does not and overall creates an impression that resembles that of a native application.

AleksIS PWA features

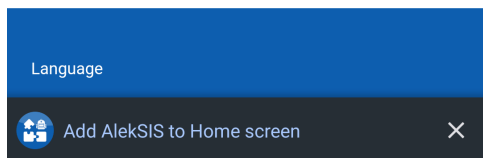
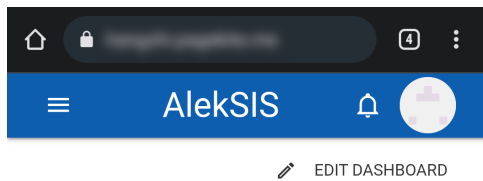
The AleksIS PWA offers the following features (not all available on all platforms):

- Installable and displayable in a separate window
- Caching and serving, if given page cannot be accessed, of non-interactive pages and needed assets
- Provision of an offline fallback page if wanted page cannot be accessed and there is no cached one
- Indicator whether the served page is served from the PWA cache

Installation of the PWA

The procedure to get a native feeling using the AleksIS PWA varies from platform to platform. On some, you are prompted to add AleksIS to your home screen of desktop using a popup; on others, you have to take action yourself and find the corresponding menu entry. As of the time of writing, “installable” PWAs are supported by all major platforms except Firefox Desktop and Safari Desktop which nevertheless support the other features.

Chromium-based browsers (e.g. Chromium, Google Chrome, Microsoft Edge) will usually prompt you to install the PWA by a popup on both mobile and desktop devices; for the former using a banner

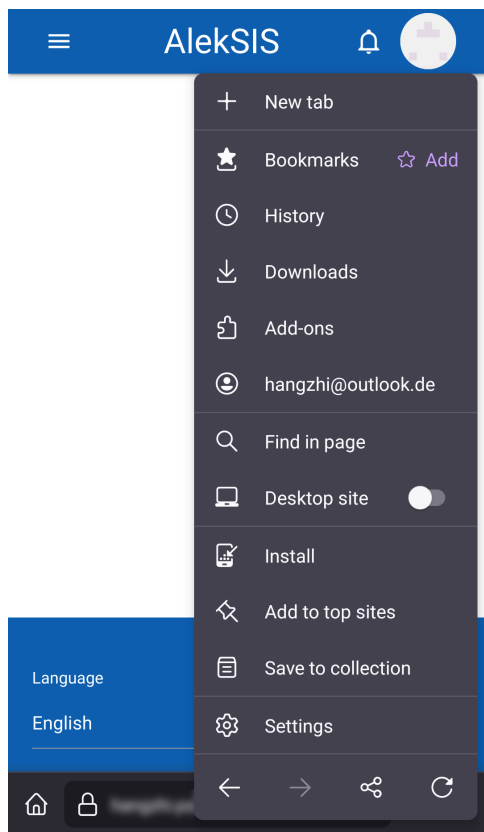


and for the latter using an appearing button in the address bar.

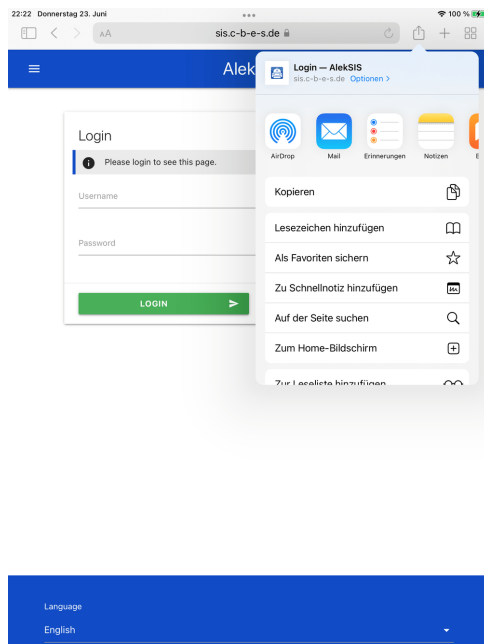


In both cases, a click on the notification is enough to start the installation process.

Firefox Mobile will also prompt you using a dot near the menu button; then Install has to be clicked.



On Safari Mobile, you need to open the share popup and click on the Add to Home Screen button.



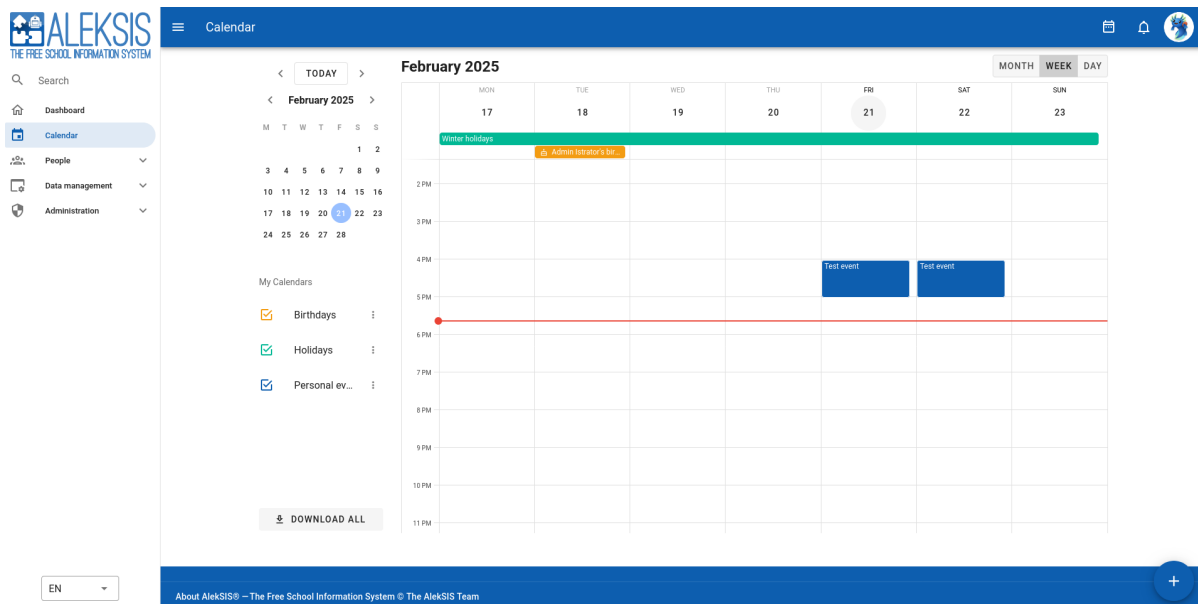
No matter what platform is used, AlekSIS can be accessed as an independent application entry, just like any other installed native application, after installation.

2.1.6 AleksIS' calendar system

AleksIS has a built-in calendar system that bundles all time-related information in one place. This includes:

- Birthdays of persons
- Holidays
- Personal events (cf. core-personal-event)

Other apps may add more calendars, such as timetables. You can access the calendar overview frontend via the Calendar menu point.

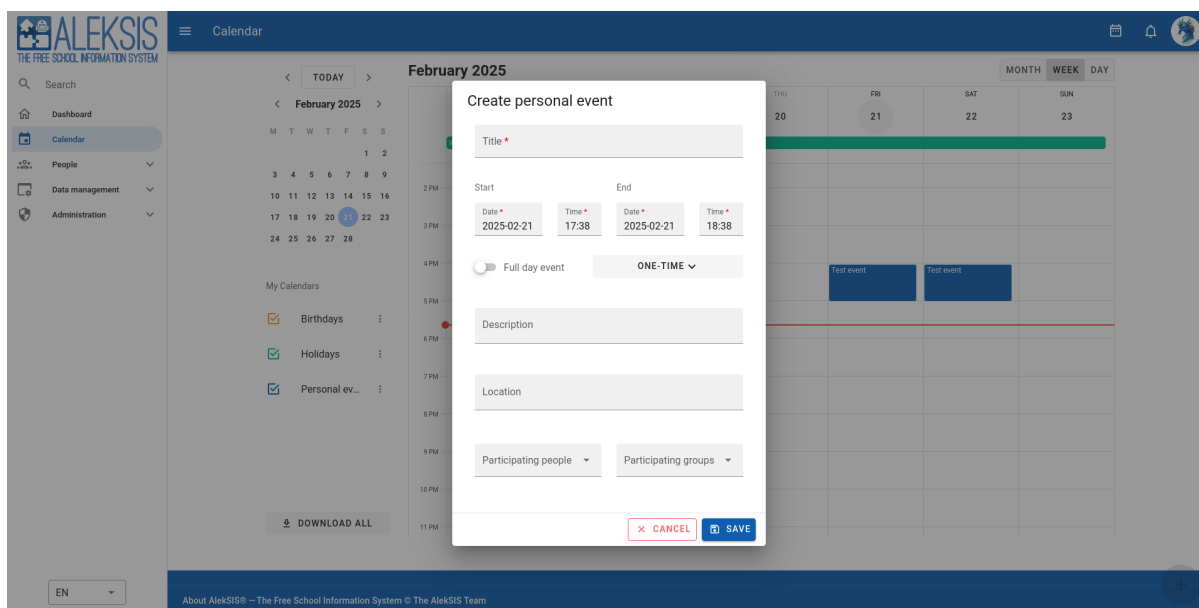


In addition to the options of changing between different views (day, week, month) and of selecting the date range you want to see, you can select which calendars should be shown using the My Calendars menu. Clicking on an event opens a dialog showing additional information about the given event, such as an event description. The calendar information shown to you is personalized: For example, users can only see the birthdays of persons they have access to.

Clicking on the button indicated by three points next to each calendar item in the My calendar menu gives you the option to download the respective calendar as an .ics file. This format is supported by nearly every calendar client (e. g. Thunderbird). Opening it with such a client allows you to view the selected calendar locally.

2.1.7 Personal events

AleksIS' calendar system allows users to create *personal events* for any purpose. This can be done via the + button on the bottom right of the calendar overview page.



Personal events must contain a start and end date or datetime as well as a title. Besides that, they can store the following data:

- **Rule for repetition:** In case you want your event to be recurring, click on the One-time button. Two fields will be shown that allow you to specify the repetition frequency and, optionally, an end date.
- **Description**
- **Location**
- **Participating people:** By default, personal events are only visible to yourself. In case you want other persons to see your event in their own calendars, you can select them here.
- **Participating groups:** All members of the groups selected here will be able to see your personal event in their own calendars.

You can edit or delete a personal event you created yourself by clicking on any instance of said event in the calendar overview page and then clicking the respective buttons in the dialog that then appears.

2.1.8 WebDAV access

AlekSIS allows users to access calendar and person data via CalDAV and CardDAV. This enables you to view this information in a local client of your choice with offline support. This includes calendars, as seen on your own calendar overview page, and all persons you are allowed to view. Persons are made accessible as contacts bundled in one address book, while each calendar corresponds to a calendar feed. All information is provided read-only. Syncing local changes back to AlekSIS is thus not possible.

You can configure CalDAV and CardDAV access in any client that supports auto discovery, such as Thunderbird. In order to gain access, use the domain of your AlekSIS instance as location and authenticate yourself with your AlekSIS credentials.

2.2 Timetables and substitution plans with Chronos

2.2.1 Finding information about the timetable

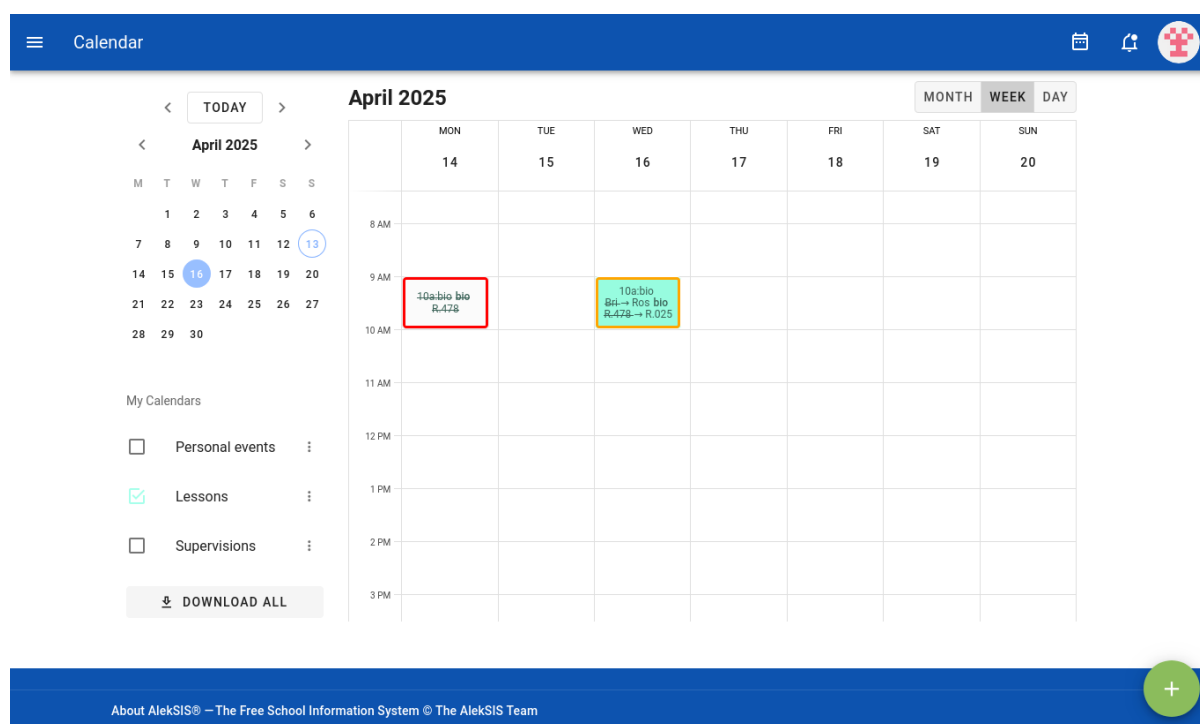
Timetables are provided in several ways in AleksIS. Using the different views, it is possible to both find information about the regular lessons taking place, as well as up-to-date information generated from most recent information, e.g. substitutions, cancellations, or extra lessons.

Viewing the personal timetable

One of the most useful features is the ability to see your daily or weekly timetable at a glance – with all recent information included. This personal timetable is provided in your personal calendar. Just open *Calendar** in the menu and tick the box next to the *Lesson* calendar.

How does the timetable look?

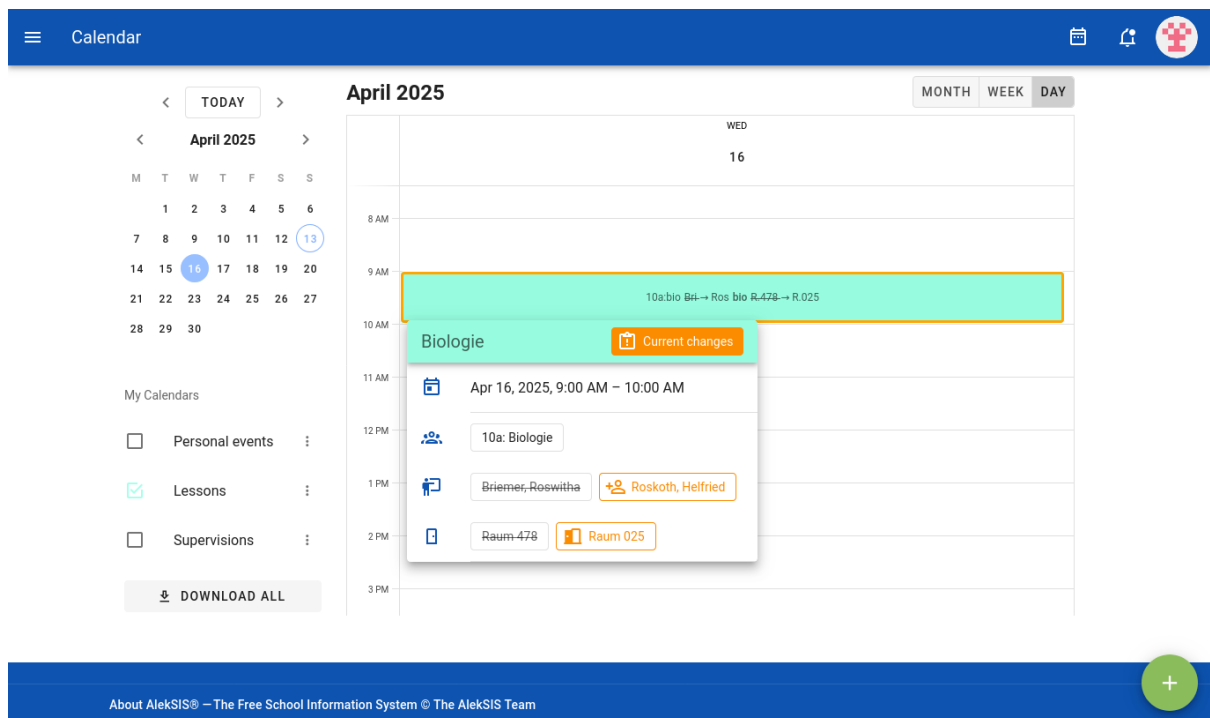
In the timetable, you can see the single lessons. By default, they are showing the lessons regularly taking place. Sometimes, they include current changes like substitutions or cancellations.



The colours of the lessons are defined by the subject of the lesson. In addition, the following highlighting is done for current changes:

- Substituted lessons are decorated with a red border, and changes in the lesson data are struck through and replaced with the new information
- Cancelled lessons have their background removed and everything is struck through
- Remarks from the substitution plan are added in italics underneath the lesson information

For more details on single lessons, just click on them for the following dialog:

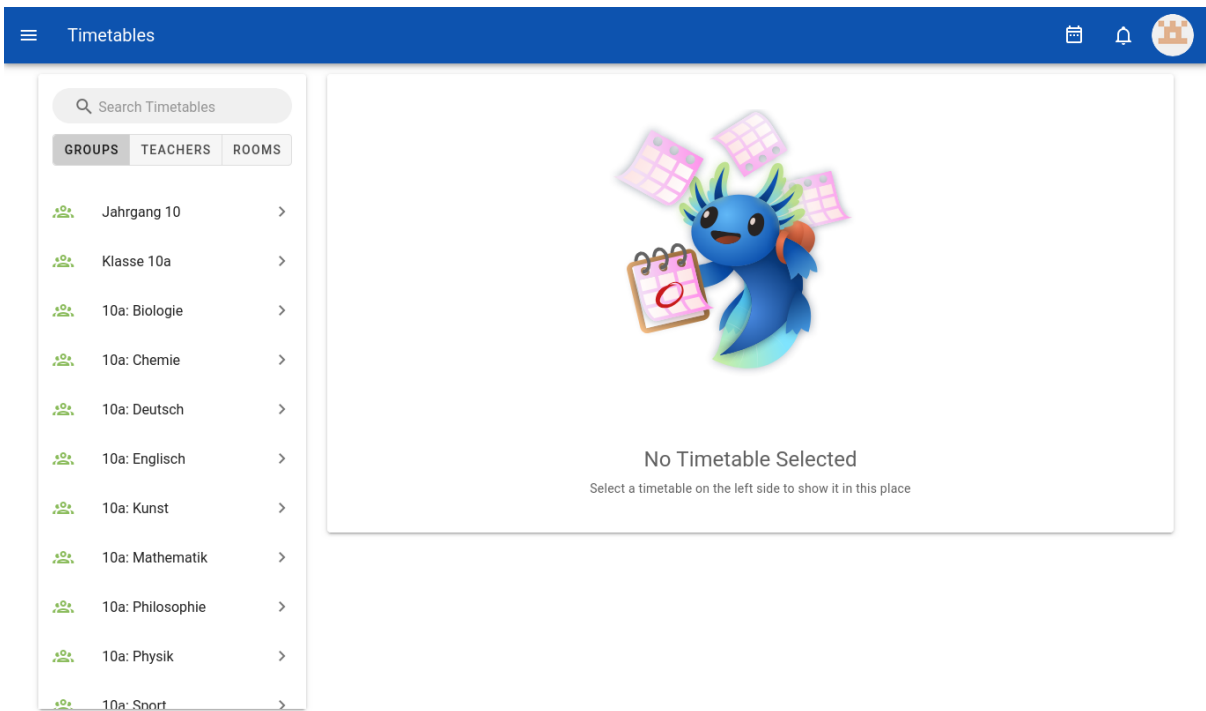


Other timetables

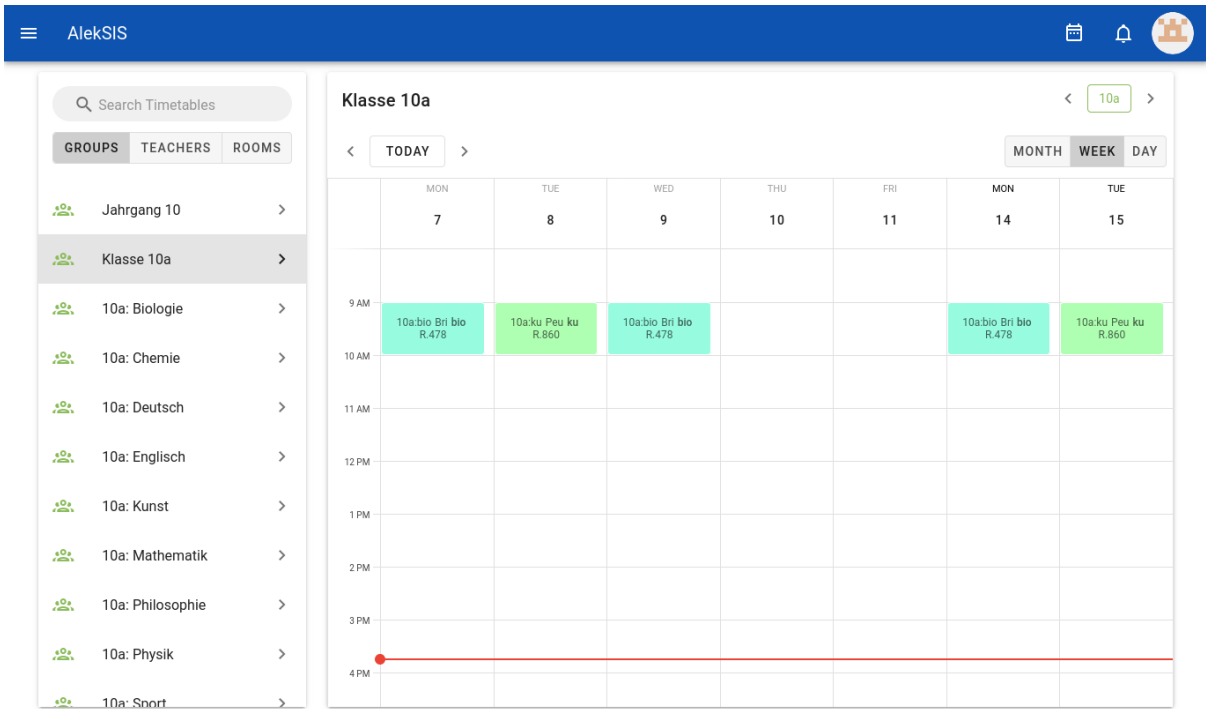
Using the menu entry *Timetables* → *Timetables* will give you access to all timetables of your institution (or whatever your administrator allowed you to see). It's divided into two main sections: On the left side, you can select the timetable you want to see. You can switch between *Groups*, *Teachers*, and *Rooms*. Using the search field above the switch, you can filter the list by their names.

It's divided into three main sections:

- Teachers
- Groups
- Rooms



If you open one of these timetables, you see the current timetable in form of a calendar. Use the calendar controls to move forward or backwards in time. Additionally, you can use the arrows on the top right to move forward/backwards between different timetables in the list.



2.2.2 Substitution plan

In addition to the calendar, substitutions can be displayed as a traditional substitution plan. The substitution plan lists all changes made to the regular plan, on a daily basis, as a table.

View in AleksIS

In AleksIS, the substitution plan is available from the menu under *Timetables* → *Substitutions*. This function provides a browsable list, where substitution plans can be navigated by day.

Substitutions

Monday, April 14, 2025

PRINT

Affected teachers

Affected groups

Search

Groups	Time	Teachers	Subject	Rooms	Notes
10a:bio	9:00 AM – 10:00 AM	Bri	bio	R.478	Cancelled Work at home

Rows per page: All 1-1 of 1

< 2025-04-14 >

Print view

The substitution plan can also be printed (as a PDF file), to make hardcopies, or to display it on digital signage.



Substitutions

April 13, 2025, 3:47 p.m.

Monday 04/14/2025

Affected teachers Bri

Affected groups 10a:bio

Groups	Time	Teachers	Subject	Room	Notes
10a:bio	9 a.m.–10 a.m.	Bri	bio	R.478	Cancelled <i>Work at home</i>

Tuesday 04/15/2025

Groups	Time	Teachers	Subject	Room	Notes
No substitutions available.					

2.2.3 Notifications about current changes

Users can get notifications about current changes to their personal timetables. If the system administrator has enabled the feature, they are sent automatically.

The notifications are sent as standard AleksIS notifications. In [Notifications](#) (page 8), you can find more information about the available channels and your configuration options.

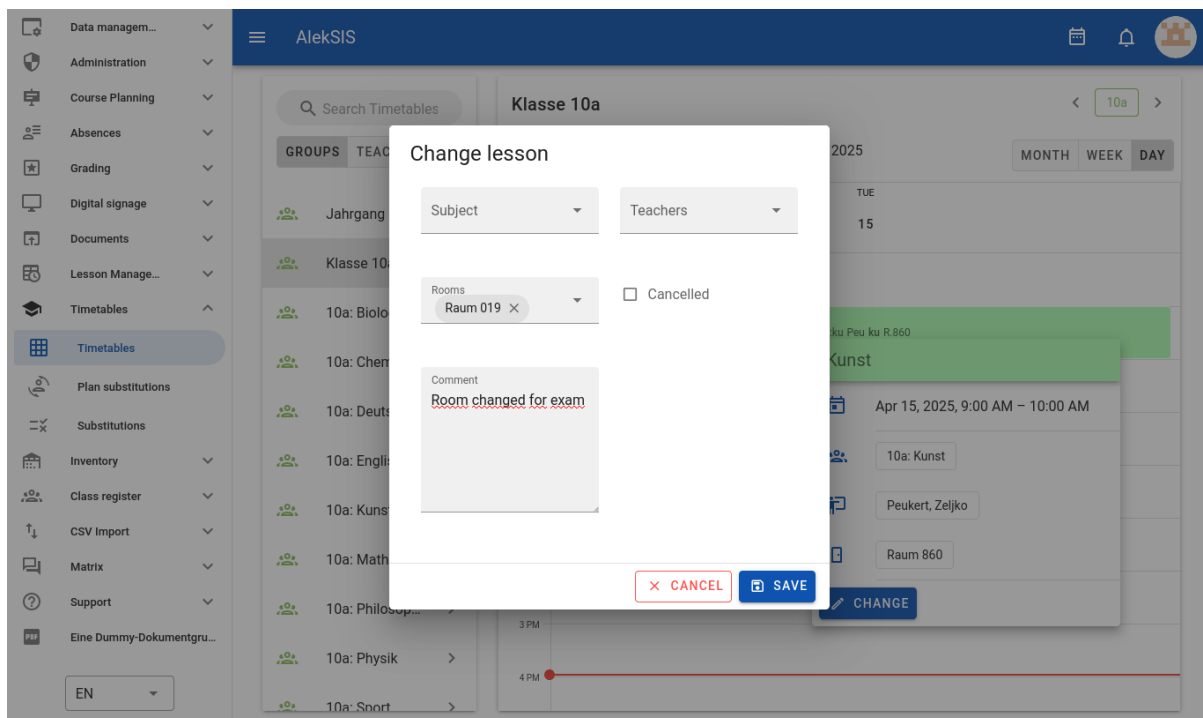
2.2.4 Managing substitutions

AleksIS has two different ways for managing substitutions and cancellations.

Simple changing of lessons

For small changes, you can just open the timetable you want to change a lesson in, select the lesson, and click the *Change* button. The dialog then allows you to change the following data for a single lesson:

- Subject
- Teachers
- Rooms
- Cancellation
- Comment

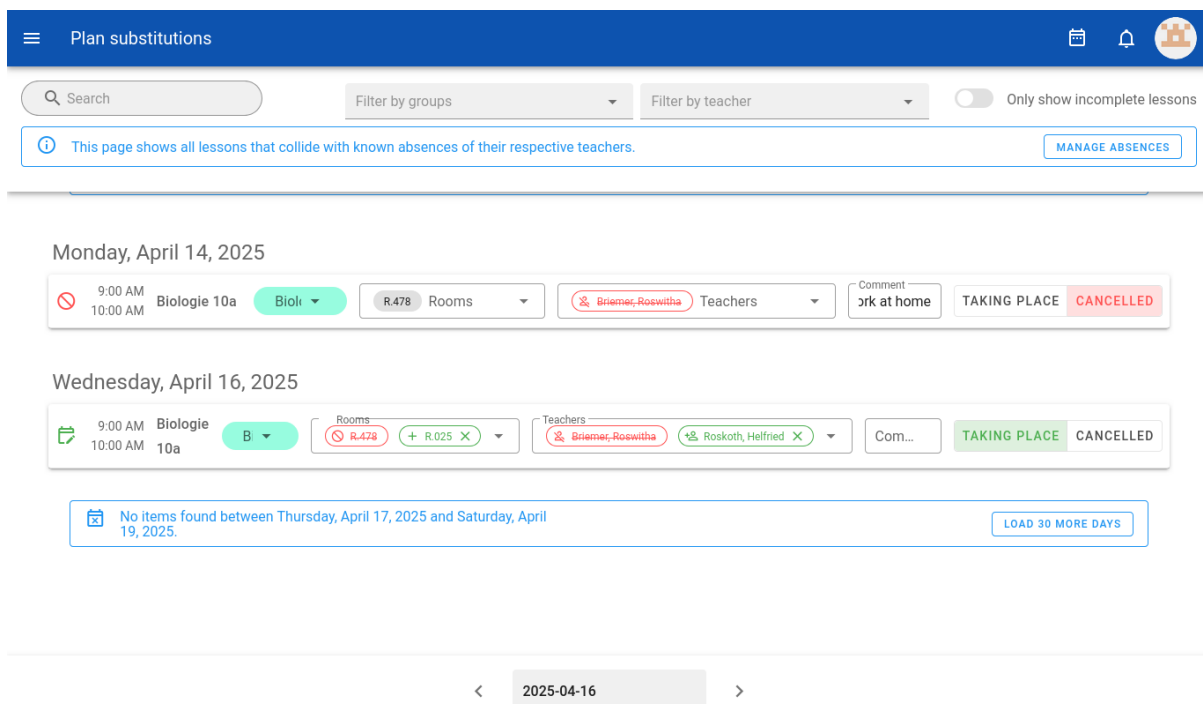


Planing based on absences

Additionally, you can plan substitutions based on teacher absences.

The first step is creating absences of teachers at *Absences* → *Absences* (a link leading to this page is also including in the substitution planning frontend).

Then you can see all affected lessons at *Timetables* → *Plan substitutions* and assign new teachers, subjects, and/or rooms. It is also possible to cancel the lesson. Students and teachers then can see the changed data and also will be notified, if activated.



Especially helpful is the select for substitution teachers. It will show you which teachers are

available at the time and teach the same subject.

The screenshot displays the 'Plan substitutions' interface. At the top, there's a search bar and filters for 'Filter by groups' and 'Filter by teacher'. A notification states: 'This page shows all lessons that collide with known absences of their respective teachers.' Below this, the calendar shows lessons for Monday, April 14, 2025, and Wednesday, April 16, 2025, both featuring 'Biologie 10a' in room R.478. A list of teachers is provided with checkboxes and status indicators: 'Good fit' (green checkmark) for Briemer, Roswitha; Dippel, Janet; Heß, Willy; and Roskoth, Helfried; and 'Doesn't teach subject' (orange triangle) for Austerhülle, Domenico and Bender, Mercedes. A 'MANAGE ABSENCES' button is located at the top right. At the bottom, a date selector shows '2025-04-14'.

2.3 Digital class register

2.3.1 Concept of Alsijil and overview about functionality

AleksIS provides a privacy-compliant online class register solution. It is not simply the digital equivalent of a paper class book, although elements are adopted for easier orientation and smoother transition for teachers. For example, there is a coursebook-like view of all lessons and a list of all the students in the class. Lesson content, notes about the student and also remarks about the learning group can be entered.

However, the application uses the possibilities and therefore the advantages of a digital application. The student lists do not have to be filled in by the class teacher, but are provided automatically by the system. The timetable is also already stored.

In addition, statistical evaluation, like counting absences, is done automatically.

2.3.2 Basic data concepts

Timetable data

The class register uses the data from the timetable app. This means that timetables with all current changes such as substitutions, events and exams can be found directly in the class book.

Even basic settings such as lesson times, holidays and public holidays do not have to be entered separately in the class book, as they are managed centrally.

Lesson documentations

Three input fields are provided for the lesson content:

1. **Lesson topic:** The content of the lesson is to be noted here, if necessary with information on the material used.
2. **Homework:** In this field, the teacher can enter the homework for the next lesson.
3. **Group note:** Here, there is space for notes that concern the whole learning group, e.g. instructions, dates, or similar.

Participation status and personal notes

For each lesson, you will find a student list of the group. The following entries can be made there:

1. **Present/Absent:** Choice between present and different absence reasons, several types can be set up also for excuses, e.g. in case a student was absent due to a school event.
2. **Tardiness in minutes** 5. **Extra marks:** This item is also configurable. A selection field for missing homework or similar would be possible here. 6. **Notes:** One or multiple personal notes can be made about the student.

With the appropriate configuration, students can view all personal data concerning themselves.

2.3.3 Overview about lessons (Coursebook)

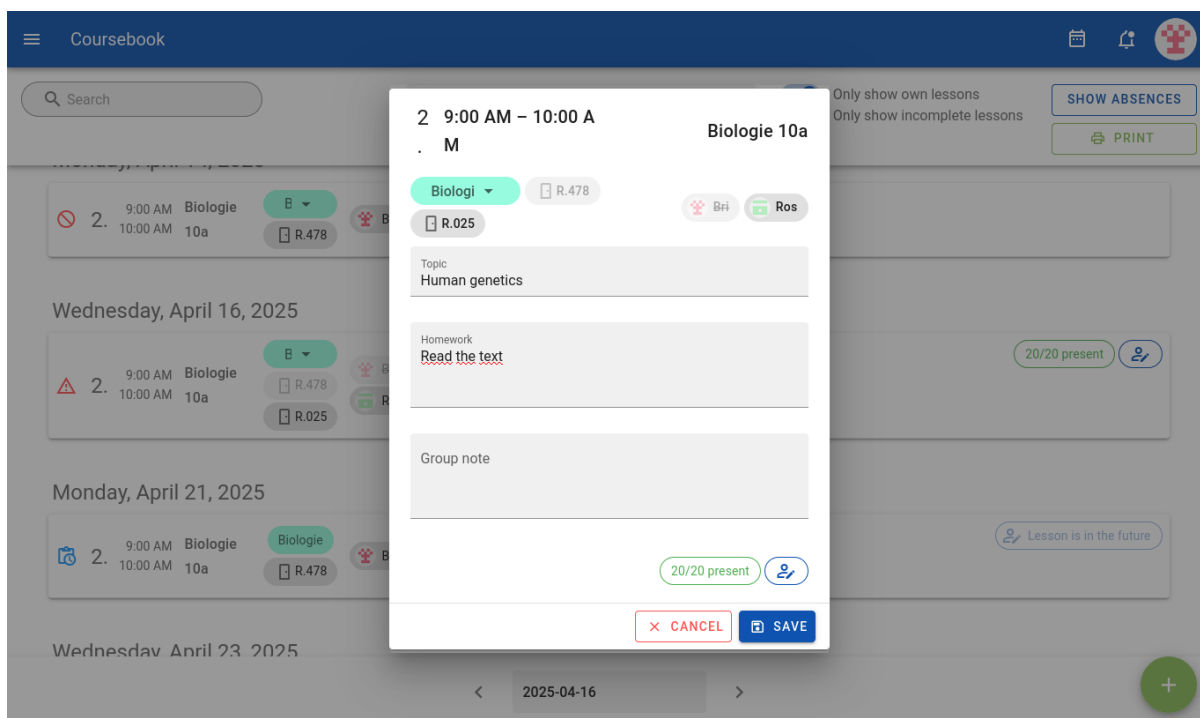
In the coursebook, all lessons for the respective user are displayed in an ongoing list. You can navigate in time by scrolling up and down, the data will be loaded automatically. It is also possible to use the date select at the bottom to select a certain point of time. To show data for a single group/course and/or only own/incomplete lessons, you can make use of the filters at the top.

Lesson details

The list already will show you most of the lesson details. To enter topics, you can directly use the text field. After leaving the field, data will be saved automatically.

The screenshot shows a user interface for lesson details. On the left, there is a green checkmark and the number '2.'. Next to it, the time '9:00 AM' and '10:00 AM' are displayed, followed by the subject 'Biologie' and '10a'. Below this, there are three filter buttons: 'B' (selected), 'R.478', and 'R.025'. To the right of these filters are two more buttons: 'Bri' and 'Ros'. A text input field labeled 'Topic' contains the text 'Human genetics'. To the right of the input field, there are two buttons: 'HW: Read the text' and 'No group note'. On the far right, there is a green button labeled '20/20 present' and a blue button with a user icon.

For homework or group notes, click on the respective field. In the dialog that will open you can enter the data and save them.



The lesson documentation is a strictly non-personal information about the contents of the lesson. It contains the topic, describing what contents were taught, and an optional homework, describing what tasks students got for the next lesson.

Everything entered here should be considered public knowledge.

Warning

Never add any personal information to the lesson documentation.

Students

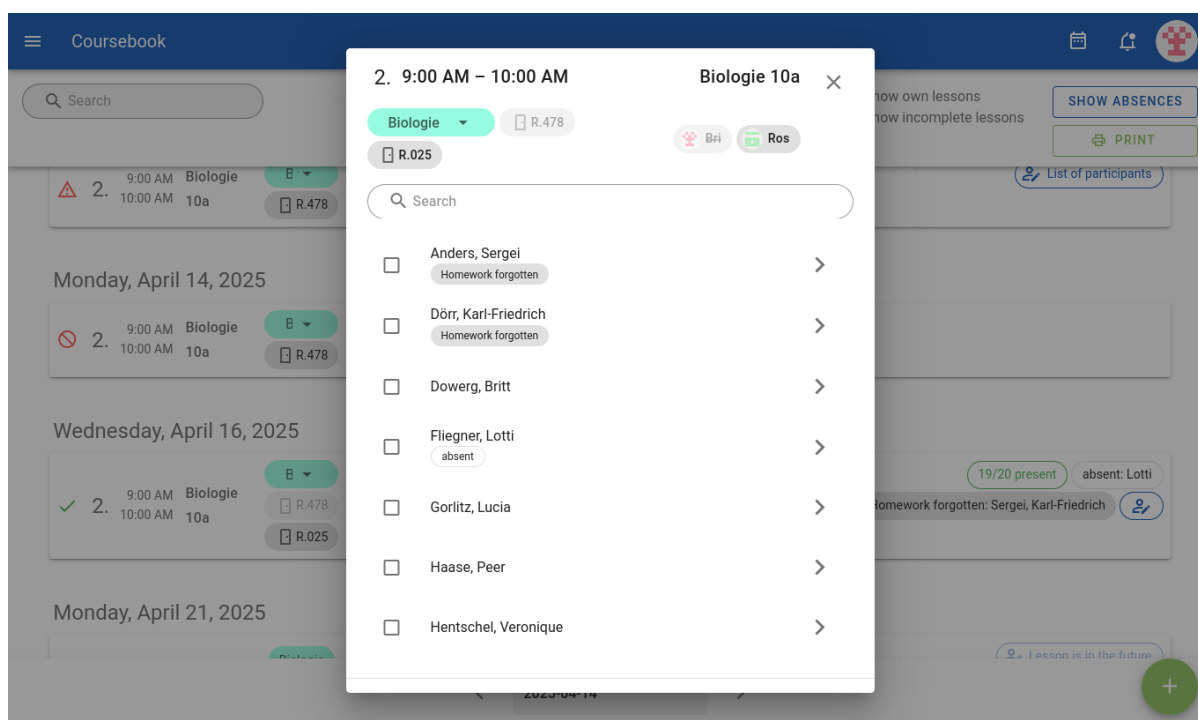
On the right side of each lesson, there is an indicator showing student data. If there are no data entered for the lesson, it just will show the button *List of participants*. Clicking on it will open a dialog for managing students.



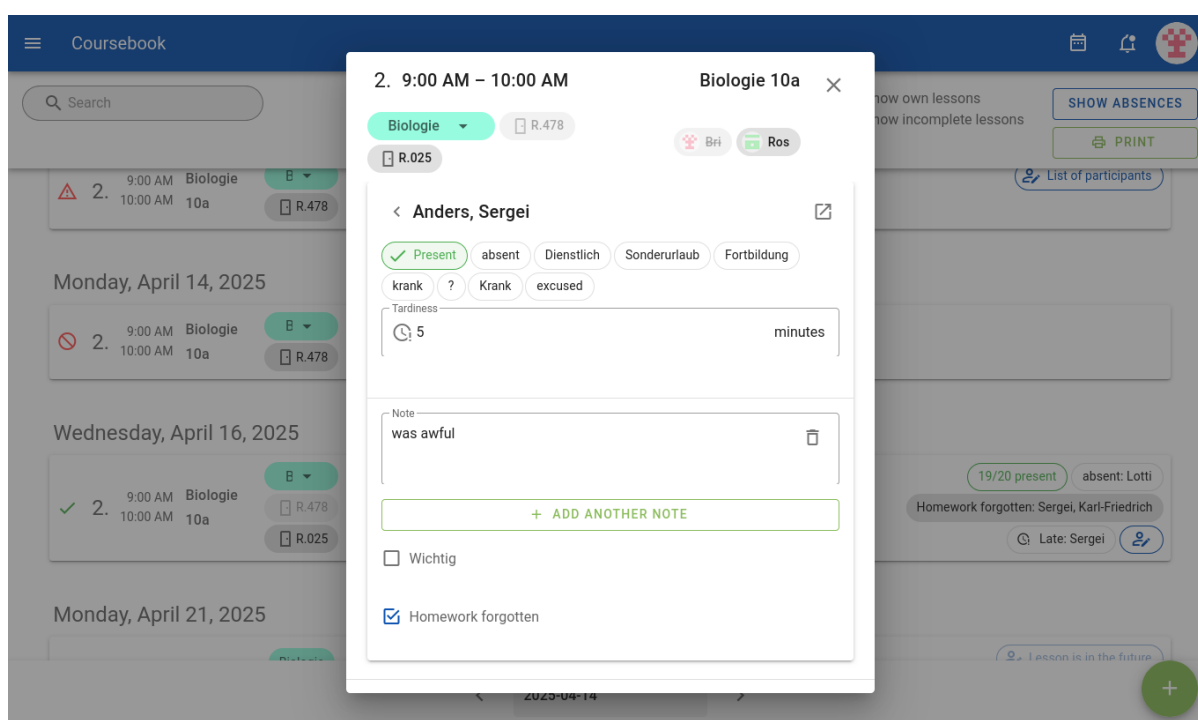
If there is already some data about students in this lesson, the indicator will show the number of students that are present and some details like absences or extra marks.



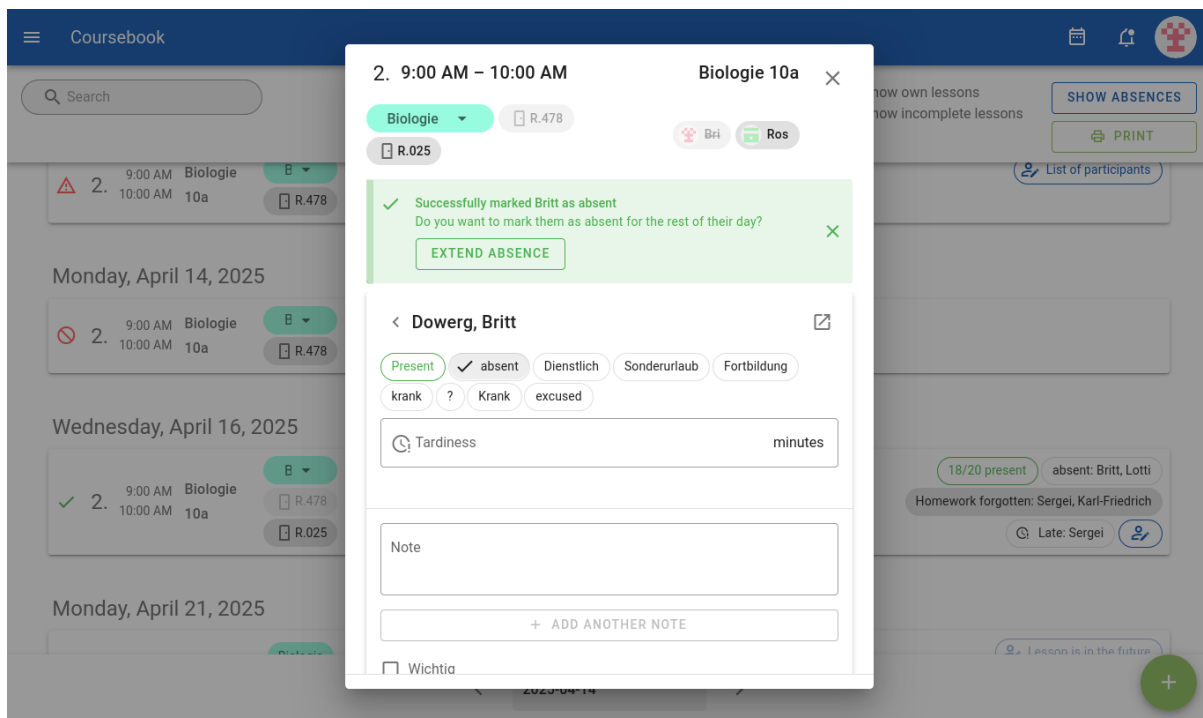
After opening the students dialog for a lesson, it shows a list of students that should be in this lesson. If there are already information about absences from other sources, students will be already marked.



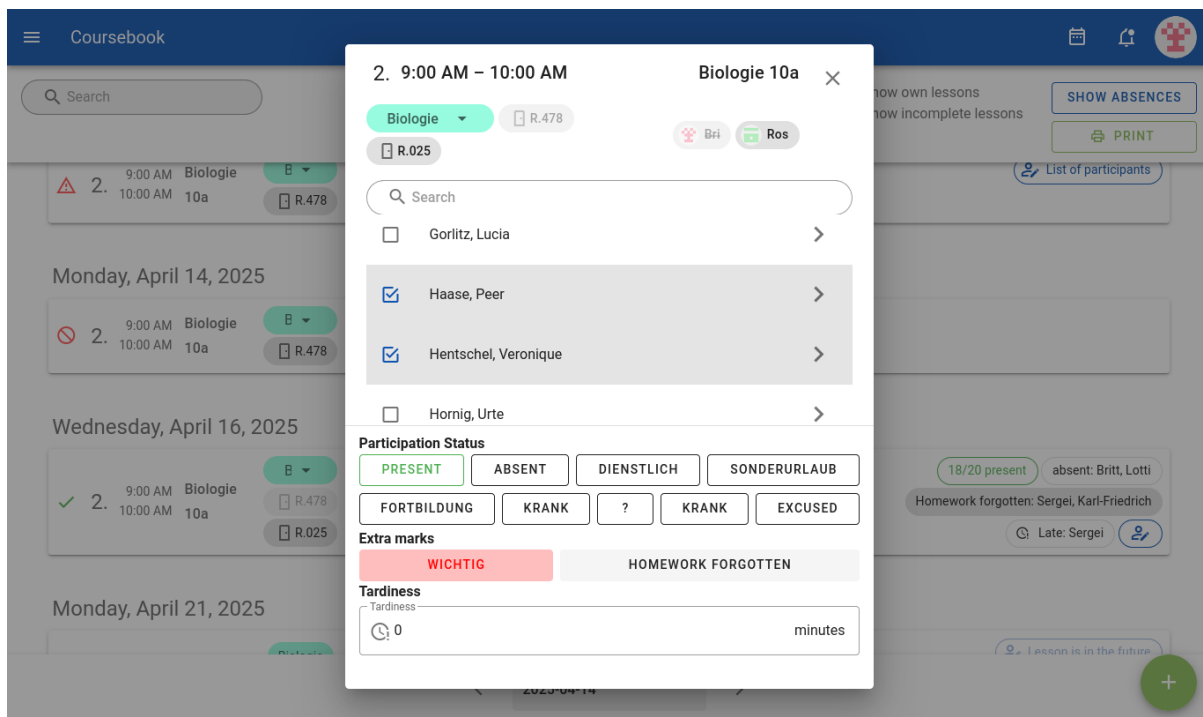
By selecting one person, you can open a detailed page about the student in this lesson. Here, you can change the participation status (from present to different absence reasons), enter tardiness, write personal notes, or set extra marks. If you open the dialog during the lesson, it will also offer you the current time as tardiness. Everything will be saved automatically.



Often students are not only absent in one lesson, but also in the following. Therefore the class register will offer you to extend the absence to the whole day.



Using the check boxes on the left side, you also can select multiple students at once to mark them as absent, late, or set an extra mark.



2.3.4 Overview about absences

Also on the coursebook page there is a large button *Show absences*. Similar to the list of lessons, it will show you a ongoing list of absences. It is possible to directly change the status of a student by clicking at one of the absence reasons.

Coursebook

Q Search

Filter for group and course

☒ Only show own lessons
☐ Only show incomplete lessons
☒ Only show lessons with absent participants

SHOW DOCUMENTATIONS

18/20 present

absent: Britt, Lotti

Homework forgotten: Sergei, Karl-Friedrich

Late: Sergei

2. 9:00 AM Biologie

10:00 AM 10a

B

R.478

R.025

Bri

Ros

☐ Dowerg, Britt

Present

absent

Dienstlich

Sonderurlaub

Fortbildung

krank

?

Krank

excused

☐ Fliegner, Lotti

Present

absent

Dienstlich

Sonderurlaub

Fortbildung

krank

?

Krank

excused

2025-04-16

+

This view also supports the selection of multiple absences to change their status together.

Coursebook

Q Search

Filter for group and course

☒ Only show own lessons
☐ Only show incomplete lessons
☒ Only show lessons with absent participants

SHOW DOCUMENTATIONS

Mark 2 selected participants as

PRESENT

ABSENT

DIENSTLICH

SONDERURLAUB

FORTBILDUNG

KRANK

?

KRANK

EXCUSED

18/20 present

absent: Britt, Lotti

Homework forgotten: Sergei, Karl-Friedrich

Late: Sergei

2. 9:00 AM Biologie

10:00 AM 10a

B

R.478

R.025

Bri

Ros

☒ Dowerg, Britt

Present

absent

Dienstlich

Sonderurlaub

Fortbildung

krank

?

Krank

excused

☒ Fliegner, Lotti

Present

absent

Dienstlich

Sonderurlaub

Fortbildung

krank

?

Krank

excused

2025-04-16

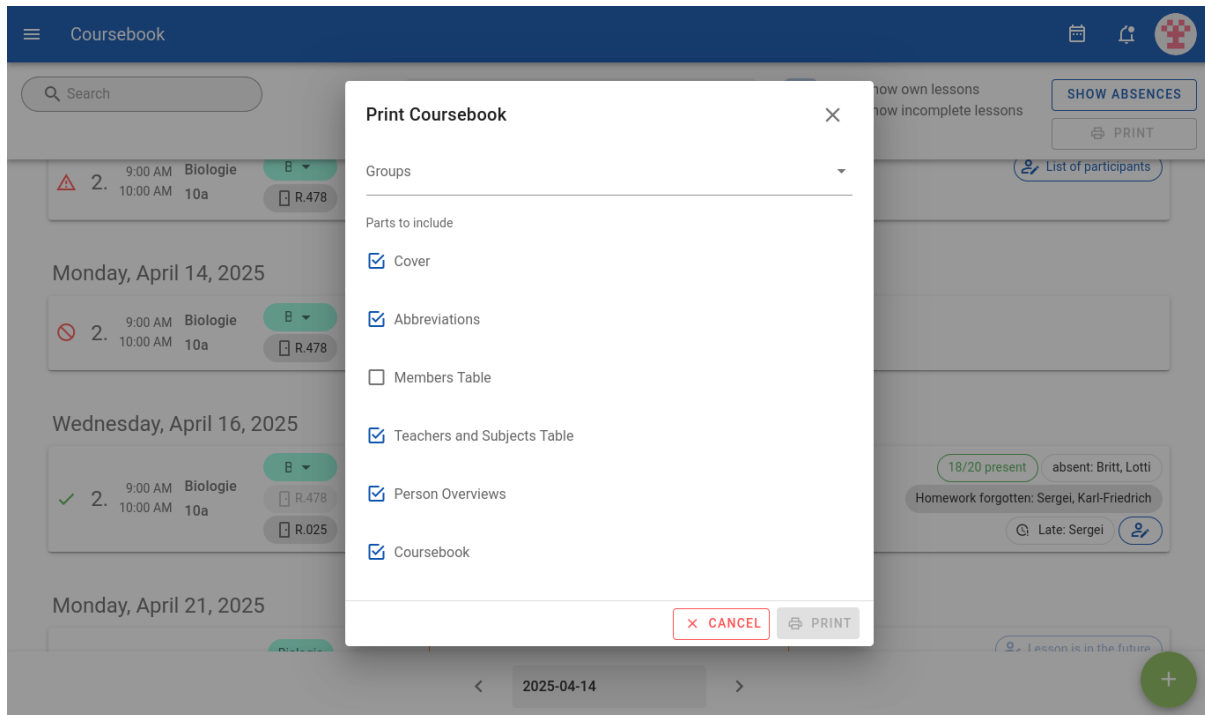
+

2.3.5 Statistics

To show statistics about groups or single students, there are widgets on the respective group or person pages.

2.3.6 (Archive) printout

Via the button *Print* on top of the coursebook page, extensive documentation can be generated for one/multiple groups. In the dialog that will open, you can configure the printout. This class register will be generated as a printable PDF file.



It contains a list of the lessons taught staggered according to timetable periods, a collection of all the students' absences and personal notes as well as all lesson contents.

Note

This mechanism can be used for archival purposes required by law.

2.4 Using seating plans

2.4.1 Managing seating plans

A seating plan is always linked to a combination of Group and Room. Additionally, a Subject can be specified. When a group, e.g. a class, has a seating plan without a subject, all subgroups, e.g. a course of that class, will inherit that plan, as long as they do not have their own seating plan. More specific relations precede more general ones, so if there exists a plan for group 5a and one for group 5a with subject English, the latter will be shown. All seating plans can be viewed, edited and created under *Seating plans* in the navigation.

Seating plans

+

CREATE SEATING PLAN

Name	Edit	Delete
5a (5a) (2021/2022) - Nw1 (Ang. Naturwissenschaften 1) - Aula (Aul)	BEARBEITEN	LÖSCHEN
5a (5a) (2021/2022) - Ag (Arbeitsgemeinschaft) - Biologie Übungsraum (BiÜ)	BEARBEITEN	LÖSCHEN
5a (5a) (2021/2022) - None - Biologie Hörsaal (BiH)	BEARBEITEN	LÖSCHEN
5a: Mu (5a-Mu) (2021/2022) - Mu (Musik) - Refektorium (Rto)	BEARBEITEN	LÖSCHEN
5a: Bi (5a-Bi) (2021/2022) - None - Biologie Hörsaal (BiH)	BEARBEITEN	LÖSCHEN

To create a new seating plan, click on Create seating plan. Select a group, subject and room and continue to the seat arrangement page. There you can use drag-and-drop to place available persons in the grid at the bottom. Extend the size of the plan, if necessary, using the expand buttons at each edge.

<

ZURÜCK

Edit seating plan

Fach

Nw1 (Ang. Naturwissenschaften 1)

Raum

Aula (Aul)

SICHERN

Available persons

Alexander, Angela

Colon, Maria

Drake, Nicholas

Hall, Chelsea

Howard, Elizabeth

Jackson, Lisa

Jimenez, Amanda

Nixon, Kelly

Smith, Carlos

Steele, Marcus

Lane, Lindsey

Hebert, Julie

Jones, Barbara

Seating plan

Montoya, Michael

Taylor, Benjamin

Morrow, Jason

Jackson, Don

Moreno, Allison

McPherson, Gregory

Duncan, Christopher

Elliott, Caroline

Harris, Michael

Gonzales, Billy

Miller, Olivia

Johnson, Anne

Seating plans can be viewed by clicking on their name in the overview, where they can also be edited or deleted.

32

Chapter 2. End-user handbook

< ZURÜCK Seating plan: 5a (5a) (2021/2022) - Nw1 (Ang. Naturwissenschaften 1) - Aula (Aul)

BEARBEITEN LÖSCHEN

 Montoya, Michael	 Taylor, Benjamin	 Morrow, Jason	 Jackson, Don	 Moreno, Allison	 McPherson, Gregory
 Duncan, Christopher	 Elliott, Caroline	 Harris, Michael	 Gonzales, Billy	 Miller, Olivia	 Johnson, Anne

2.4.2 Integration with class register









When the class register functionality of AleksIS is used, there will be a tab for seating plans on lesson documentation pages.

STUNDE
PERSONEN
SEATING PLAN
MORE

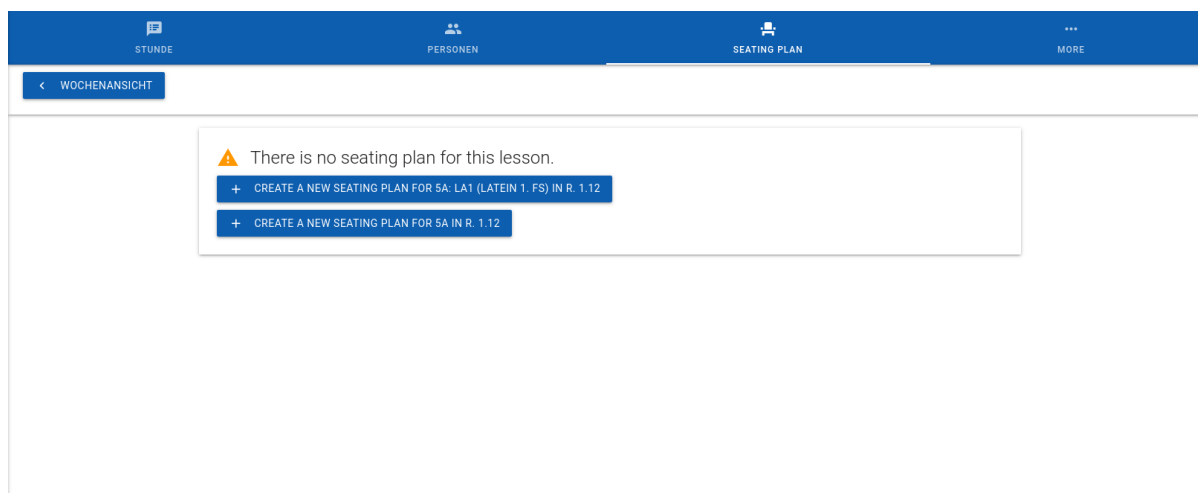
< WOCHENANSICHT

Seating plan for 5a: Bi (5a-Bi) (2021/2022) in Biologie Hörsaal (BiH)

EDIT SEATING PLAN

 Simmons, Chloe	 Smith, Michelle	 Williams, Brenda	 Silva, Toni
 Scott, Dorothy	 Underwood, Jeffery	 Smith, Gregory	 Ward, Cynthia

If there is no seating plan for a lesson, i.e. no one for that specific combination of Group, Room and Subject, no one for the combination of Group and Room and if none of the parent groups has one, options for creating a seating plan are displayed.



2.5 Time-based documents with Resint

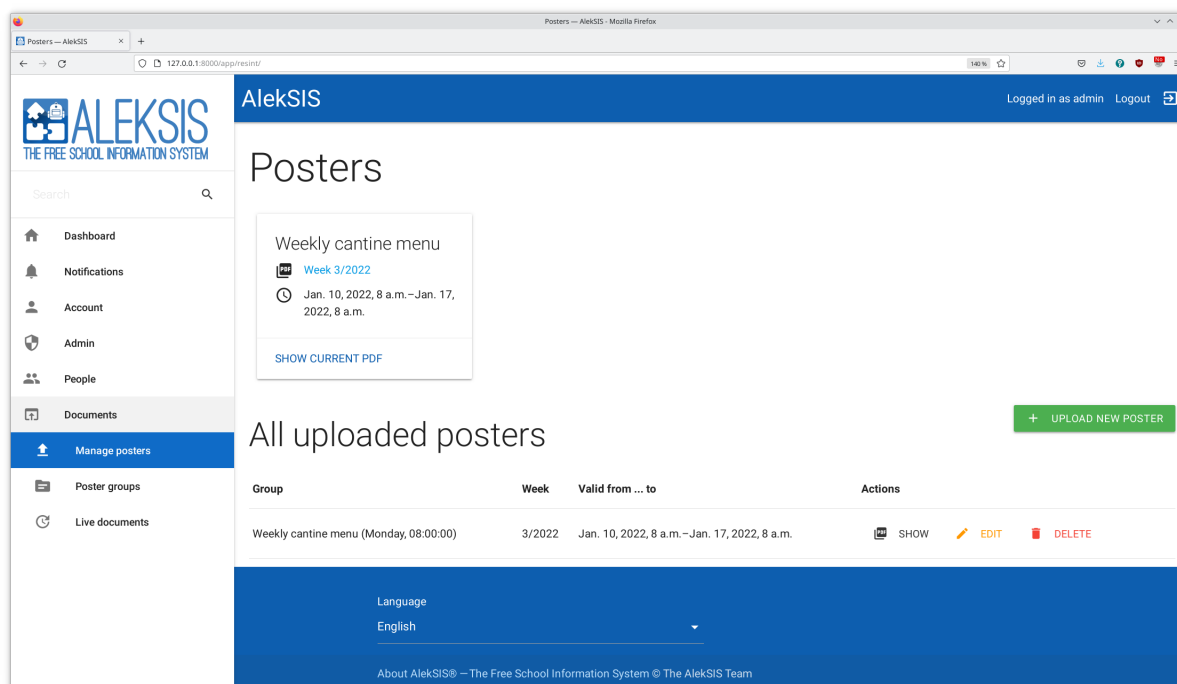
2.5.1 Publishing posters

Posters are documents that can be manually supplied in a time-based manner.

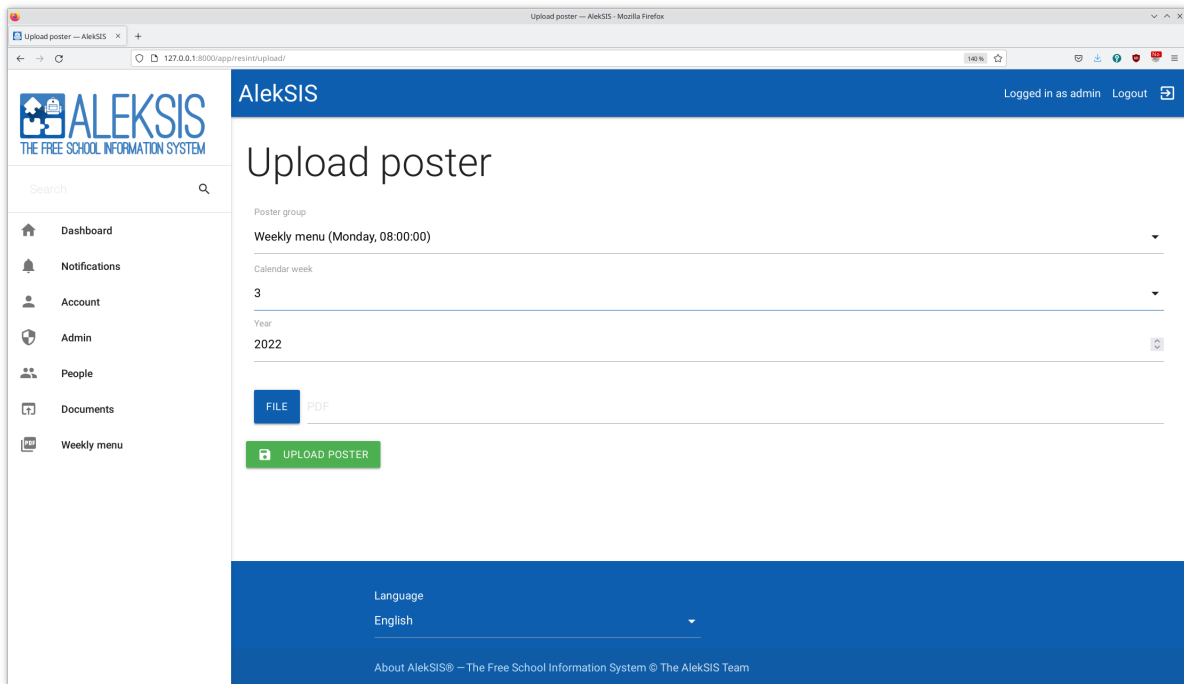
Poster files can be uploaded for a defined time period, and AlekSIS will then deliver the currently valid version under a stable menu item and URL.

Posters are organised in poster groups, which can be created by administrators. Users can then upload PDF files which shall be published for each time period.

Users who have the permission to upload posters for at least one poster group can manage their posters from the main menu under the *Documents* → *Manage posters* menu item.



Using the *Upload poster* button or the *Edit* action for an already uploaded poster will open the dialog for managing a document for one time period.

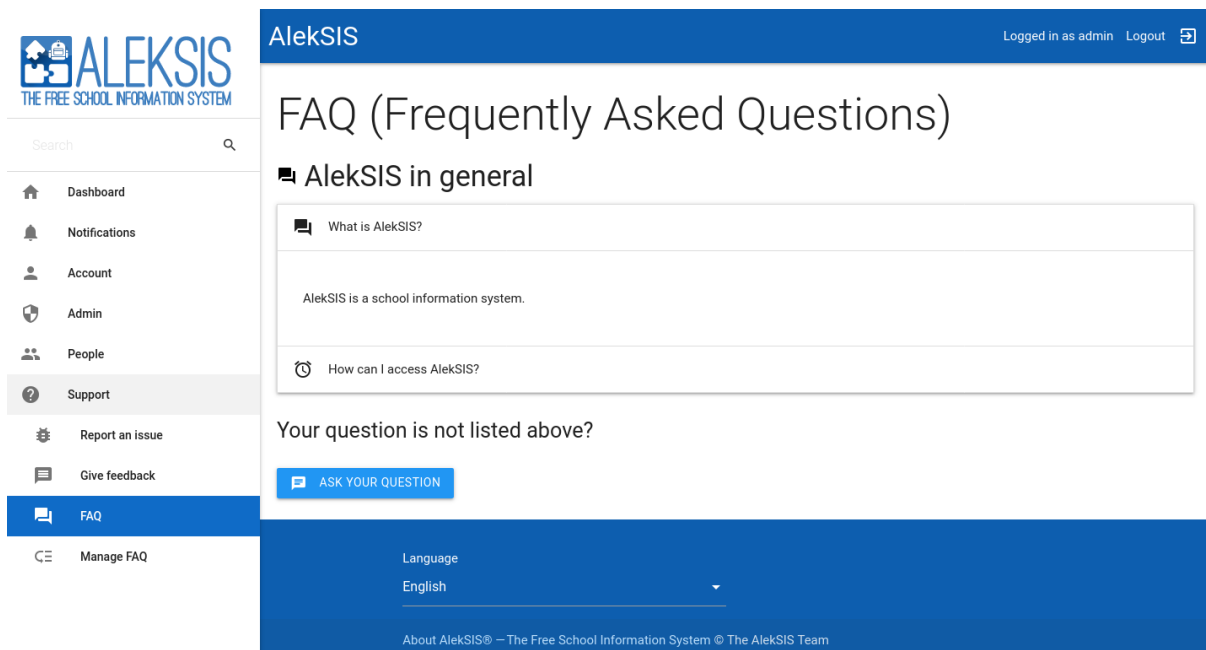


After uploading, AleksIS will deliver the uploaded file during the week selected in the form.

2.6 Get support via Hjelp

2.6.1 Reading Frequently Asked Questions

In the *Support* menu, administrators can add Frequently Asked Questions concerning the AleksIS instance, the local network, or something completely different. These questions and their answers are accessible to all users under the *Support* → *FAQ*. If the system administrator enabled it, that might be also possible without login.



If the question is not answered, users have the possibility to ask a question using a dedicated form.

2.6.2 Reporting issues

Using *Support* → *Report an issue*, users can report issues e. g. with AleksIS or the school network. To allow operators to fastly narrow down the issue, there is a three-level category system filled individually for each AleksIS instance. In addition to the category, the form asks for a one-line description and then a detailed description with additional information. When reporting issues, please try to be as specific as possible – it will help your administrators. On top of the entered information, the system will send your username together with the data.

2.6.3 Giving feedback for AleksIS

On the page *Support* → *Feedback*, we provided a feedback form helping us to improve AleksIS. If you fill out the form, it will be send to your system administrator which can review it and forward it to us, if necessary.

We ask you for the following information:

- **Ratings:** The following categories can be rated with one to five stars
 - Design of the user interface
 - Speed
 - User friendliness
 - AleksIS in general
- **Free text fields:** In the following fields, you can enter additional information
 - What do you like? What would you change?
 - What else do you want to tell us?
 - What do you think should be added to AleksIS?

AleksIS

Logged in as admin Logout

Feedback

Please give us detailed and honest feedback so that we can make AleksIS even better!

Design of the user interface *	★	★	★	☆	☆
Speed *	★	★	★	★	☆
User friendliness *	★	★	★	☆	☆
AleksIS in general *	★	★	★	★	★

What do you think should be added to AleksIS?

What do you like? What would you change?

What else do you want to tell us?

SEND FEEDBACK ➤

ADMINISTRATION HANDBOOK

3.1 Setup of the AlekSIS instance and the AlekSIS core

3.1.1 Concepts of the AlekSIS core

The AlekSIS core provides functionality and data models as a base for all apps.

The concept of school terms

In AlekSIS, mostly everything is based on school terms. A school term is a defined time range which can be used to link data to it. Typically, such data are learning groups, timetables or class register records. Although their name suggests it, school terms don't have to be a half or a full year. They should depend on the way how you organise data in your institution.

For example, if you issue reports at the end of every half year, a half year would be a good time range for your school terms because the class register statistics are evaluated for school terms.

Anyway, you should create a school term before you start to import or create other data entries like persons or groups.

Manage school terms

You can manage the school terms if you login with your admin account and open the menu entry `Data management` → `School terms`. There you can find a list of all school terms and buttons to create, edit or delete school terms. Please be aware that there can be only one school term at time and each school term needs a unique name.

The currently active school term can be selected using the `Active school term` menu in the app bar. It defaults to the school term that includes the current date. When viewing school term related data such as class register statistics, the currently set active school term determines the subset of data that is shown.

The concept of persons

The person model is designed to save all the data of students, teachers, guardians and any other persons of the school society. It tracks information like the following:

- Full name
- Short name
- Sex
- Date of birth
- Contact details (phone numbers, email)

- Address details
- Photo
- Relation to guardians
- Primary group (e. g. a class or a tutor group, cf. [The concept of groups](#) (page 40))

Except for the name, all data points are optional, so you can decide on your own (and based on your local data protection laws) which data should be included in AlekSIS.

There are two important things you should know about persons:

- **Persons are not automatically users:** That means that persons can be linked to a user account including things like a password and the ability to login, but they don't have to be. For example, your AlekSIS instance could save the data about parents, but you don't want them to login: In this scenario, the guardians are available as persons **without** user accounts.
- **Persons are not linked to school terms:** As persons like students are not only at the school for one school term, persons are not linked to school terms.

Manage persons

The main method to manage persons is the view under People → Persons. To add person to groups, you have to open the respective group and set the person as a member or an owner.

The concept of groups

The AlekSIS groups are a universal way to organise persons in collections like classes, courses, tutor groups, clubs, or any other division you could imagine. They track the following data:

- Group name and short name
- Owners (e. g. class or course teacher(s))
- Members (e. g. students)
- Parent groups (e. g. a class could be a parent group for a course)
- Group type (e. g. class, course, club, etc.)

In contrast to persons, groups are supposed to be **linked to school terms** (but they don't have to be). For example, the composition of a class or a course varies from school term to school term. In order to archive historical data according to local laws, these groups have to be separated which is solved by linking them to a school term.

Manage groups

Groups are managed on the page People → Groups. There you can search, view, create, change and delete groups.

Manage group types

You can manage your local group types by opening the menu entry People → Group types as an admin user.

Import school terms, persons and groups from other data sources

When AleksIS is not your single data source, all these data can be imported from other sources. You can find further information in the respective integration apps.

The concept of rooms

The rooms model allows you to manage places (e. g. actual rooms, sports fields) that can then be linked to other location-based entities (e. g. lessons). It tracks the following data:

- Room name and short name

Manage rooms

Rooms are managed on the page `Data management → Rooms`. There you can search, view, create, change and delete rooms.

Import rooms from other data sources

When AleksIS is not your single data source, rooms can be imported from other sources. You can find further information in the respective integration apps.

The concept of calendar feeds and calendar events

In AleksIS, every category of time-related information is organised in a separate calendar feed. Each calendar feed is populated with events based on a given logic. Calendar event instances are used to track the required information. In some cases, event data is filled based on data that is already present in instances of another model (e. g. birthdays of persons). If not extended, these calendar events can track the following data:

- Start and end date/datetime
- Timezone
- Recurrence rule

In AleksIS' core, there are three predefined calendar feeds:

- Birthdays (of persons)
- Holidays
- Personal events

The events contained in calendar feeds can be viewed on the `Calendar` page. Users are able to select which feeds they want to see in their personal calendar overview.

Configure calendar feeds

You can configure calendar-related settings via the `Calendar` tab of the `Administration → Configuration` page. This includes:

- The first weekday to appear on the calendar frontend
- The colors assigned to the birthday, holiday and personal event feeds, respectively

The concept of holidays

Holidays are used to save the time periods of (public) holidays. They are displayed as contextual information in the calendar overview and all relevant calendar views (e. g. timetables). They track the following data:

- Holiday name
- Start and end date

Manage holidays

Holidays are managed on the page `Data management` → `Holidays`. There you can search, view, create, change and delete holidays.

3.1.2 Install AleksIS

From PyPI

In this section we will install AleksIS with *uWSGI* and *nGINX* on Debian bookworm.

Filesystem locations

AleksIS will need and use the following paths:

- `/etc/aleksis` for configuration files
- `/var/lib/aleksis/media` for file storage (Django media)
- `/var/backups/aleksis` for backups of database and media files
- `/usr/local/share/aleksis/static` for static files
- `/var/cache/aleksis` for building frontend assets etc.

You can change any of the paths as you like.

Prerequisites

For an installation on a dedicated server, the following prerequisites are needed:

- Debian 12 or newer, or comparable distribution
- PostgreSQL with the `pg_rrule` extension
- Valkey (or legacy Redis)
- uWSGI
- nginx
- Python 3.10 or newer
- Node.js 18 or newer
- Some system dependencies to build Python modules and manage frontend files
- System locales for all supported languages
- The aforementioned paths

Install system packages

Install some packages from the Debian package system.

```
apt install postgresql-common
/usr/share/postgresql-common/pgdg/apt.postgresql.org.sh -y
apt install uwsgi \
    uwsgi-plugin-python3 \
    nginx-full \
    python3 \
    python3-dev \
    python3-pip \
    libpq-dev \
    yarnpkg \
    python3-venv \
    firefox-esr \
    redis-server \
    pkg-config \
    postgresql-17 \
    postgresql-17-pg-rrule \
    locales-all \
    celery
```

Create PostgreSQL user and database

Generate a secure password for the database, then create the user and database.

```
sudo -u postgres createuser -D -P -R -S aleksis
sudo -u postgres createdb -E UTF-8 -O aleksis -T template0 -l C.UTF-8_
→aleksis
sudo -u postgres psql -c "CREATE EXTENSION pg_rrule" aleksis
```

When asked, use the password generated above.

Create the directories for storage

```
mkdir -p /etc/aleksis \
    /usr/share/aleksis/static \
    /var/cache/aleksis \
    /var/lib/aleksis/media \
    /var/backups/aleksis
```

Create AleksIS configuration file

AleksIS is configured in files in `/etc/aleksis`. Create a basic configuration file for the environment defined above by opening `/etc/aleksis/aleksis.toml` with your favourite text editor and adding the following configuration.

```
static = { root = "/usr/local/share/aleksis/static", url = "/static/" }
media = { root = "/var/lib/aleksis/media", url = "/media/" }
```

(continues on next page)

(continued from previous page)

```
 caching = { dir = "/var/cache/aleksis" }
 secret_key = "SomeRandomValue"
```

[http]

```
 allowed_hosts = ["aleksis.example.com"]
```

[database]

```
 host = "localhost"
 name = "aleksis"
 username = "aleksis"
 password = "password_generated_above"
```

[backup]

```
 location = "/var/backups/aleksis"
```

[auth.superuser]

```
 username = "admin"
 password = "admin"
 email = "root@localhost"
```

Make sure you specify the correct *allowed_hosts*. It is used for things like the OpenID Connect issuer and the standard mail domain.

Generate OpenID Connect certificate

OpenID Connect needs a certificate, which you need to generate and assign required permissions.

```
openssl genrsa -out /etc/aleksis/oidc.pem
chown www-data:www-data /etc/aleksis/oidc.pem
```

Install AleksIS itself

To install AleksIS now, and run all post-install tasks, run the following commands. They will pull the AleksIS standard distribution from [PyPI](https://pypi.org)⁴ and install it into a newly created venv. Afterwards, it will download frontend dependencies from *yarnpkg*, collect static files, and migrate the database to the final schema.

aleksis is a meta-package which will install the [Concepts of the AleksIS core](#) (page 39) and all official apps. If you want to install only the AleksIS Core and your own set of apps, you can install *aleksis-core* instead of *aleksis*.

You only need to install these additional dependencies if installing the meta-package:

```
apt install libmariadb-dev libldap2-dev libsasl2-dev pkg-config
```

After that, you can install the *aleksis* meta-package, or only *aleksis-core*. The `pip3` and `aleksis-admin` commands have to be executed with the venv activated.

⁴ <https://pypi.org>


```
python3 -m venv aleksis-venv
source aleksis-venv/bin/activate
pip3 install aleksis
aleksis-admin vite build
aleksis-admin compile_scss
aleksis-admin collectstatic --clear
aleksis-admin migrate
aleksis-admin createinitialrevisions
```

Make dynamic content writable for webserver

To make AleksIS® able to write dynamic content, you need to assign permissions to the web-server user.

```
chown -R www-data:www-data /var/lib/aleksis
```

Configure uWSGI

uWSGI is an application server that will manage the server processes and requests. It will also run the Celery broker and scheduler for you.

Configure a uWSGI app by opening `/etc/uwsgi/apps-available/aleksis.ini` in an editor and inserting:

```
[uwsgi]
vhost = true
plugins = python3
master = true
enable-threads = true
processes = 20
wsgi-file = /usr/local/lib/python3.11/dist-packages/aleksis/core/wsgi.py
chdir = /var/lib/aleksis
lazy = true
lazy-apps = true
attach-daemon = celery -A aleksis.core worker --concurrency=4
attach-daemon = celery -A aleksis.core beat
```

Afterwards, enable the app using:

```
ln -s /etc/uwsgi/apps-available/aleksis.ini /etc/uwsgi/apps-enabled/
→aleksis.ini
service uwsgi restart
```

Configure the nginx webserver

First, you should get a TLS certificate, e.g. by using [Let's Encrypt](https://certbot.eff.org/instructions)⁵.

Then, create a virtual host in nginx, by editing `/etc/nginx/sites-available/aleksis.example.com`.

⁵ <https://certbot.eff.org/instructions>

```
upstream aleksis {
    server unix:///run/uwsgi/app/aleksis/socket;
}

server {
    listen 80;
    listen [::]:80;

    server_name aleksis.example.com;

    return 301 https://$server_name$request_uri;
}

server {
    listen 443 ssl http2;
    listen [::]:443 ssl http2;

    ssl_certificate /etc/letsencrypt/certs/aleksis.example.com/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/certs/aleksis.example.com/privkey.
→pem;
    ssl_trusted_certificate /etc/letsencrypt/certs/aleksis.example.com/chain.
→pem;

    server_name aleksis.example.com;

    access_log /var/log/nginx/access.log;

    location /static {
        alias /usr/local/share/aleksis/static;
    }

    location / {
        uwsgi_pass aleksis;
        include uwsgi_params;
        proxy_redirect off;
        proxy_pass_header Authorization;
    }
}
```

Enable the virtual host:

```
ln -s /etc/nginx/sites-available/aleksis.example.com /etc/nginx/sites-
→enabled/aleksis.example.com
service nginx restart
```

Finalisation

Your AleksIS installation should now be reachable and you can login with the administration account configured above.

With Docker

AleksIS can also be installed using Docker, either only AleksIS itself, or the full stack, including Valkey, using docker-compose

Full stack using docker-compose

First, install Docker and docker-compose on your system. Also install git to get the docker-compose file and image definition.

```
apt install docker.io docker-compose git
```

Now, clone the distribution repository, which contains the docker-compose file.

```
git clone https://edugit.org/AleksIS/official/AleksIS
```

By default, the current development version is checked out. Use

```
git checkout $ALEKSIS_VERSION
```

to check out the current version, by replacing `$ALEKSIS_VERSION` with the AleksIS release version number.

You should review the file *docker-compose.yml* for any environment variables you want to set, e.g. using a *.env* file. For a production instance, set at least *ALEKSIS_HOSTNAME* and *POSTGRES_PASSWORD*.

A *.env* file could look like this:

```
ALEKSIS_HOSTNAME=sis.example.org
POSTGRES_PASSWORD=VerySecretPassword
```

Note that for a testing installation without TLS, or if you use a custom port, you have to manually add something like the following to your *docker-compose.yml* to the environment of the *app* container definition:

```
- ALEKSIS_http__base_url="http://sis.example.org:8080"
```

Finally, bring the stack up using:

```
docker-compose up -d
```

AleksIS will be reachable on port 80 if you forgot to configure the environment. You are responsible for adding a reverse proxy like nginx providing TLS, etc.

3.1.3 Configuration files and format

File locations and order

AlekSIS is configured through text files in the directory `/etc/aleksis/`. You can place any file types there that are supported by the [Dynaconf](#)⁶ configuration system (INI, YAML and TOML).

Files are evaluated in alphabetical order, with later definitions overwriting earlier ones. Normally, there will be only one configuration file, but you can make up any structure you like. If you use multiple files, it might be a good idea to number them, e.g. `00_main.toml`, `01_myschool.toml`, `99_temporary.toml`.

The TOML format is recommended and is the only format described in detail in AlekSIS' documentation. For all other formats, refer to the [Dynaconf](#)⁷ documentation.

Configuration file format (TOML)

TOML file are simple text files that define variables, much like in Python (i.e. there are scalars, lists and dictionaries). AlekSIS structures its configuration by topic.

A configuration file might look like this:

```
secret_key = "VerySecretKeyForSessionSecurity"

[http]
allowed_hosts = [ "aleksis.myschool.example.com", "localhost" ]

[database]
name = "aleksis"
user = "aleksis"
password = "SuperSecretPassword"

[cacheing]
valkey = { enabled = true, address = "redis://127.0.0.1" }
```

The `secret_key` setting above defines a single value. The following `http` section defines a table (cf. a dictionary) in one way, and you can see the second form of such a table in the `valkey` setting (we could as well have defined another section and placed `enabled` and `address` below it as scalars).

This can be a bit confusing, so this documentation will explain how to configure AlekSIS on a per-feature basis.

3.1.4 Configuration options

AlekSIS provides lots of options to configure your instance.

Configuration file

All settings which are required for running an AlekSIS instance are stored in your configuration file `/etc/aleksis/aleksis.toml`.

Example configuration file:

⁶ <https://dynaconf.readthedocs.io/en/latest/>

⁷ <https://dynaconf.readthedocs.io/en/latest/>

```

# General config for static, media and secret key, required
static = { root = "/srv/www/aleksis/data/static", url = "/static/" }
media = { root = "/srv/www/aleksis/data/media", url = "/media/" }
secret_key = "Xoc8eiwah3neehid2Xi3oomoh4laem"

# Localization
[l10n]
lang = "en"
tz = "Europe/Berlin"
phone_number_country = "DE"

# Admin contat, optional
[contact]
admins = [{"AleksIS - Admins", "root@example.com"}]
from = 'aleksis@example.com'

# Allowed hosts, required
[http]
allowed_hosts = ["localhost"]

# Database for whole AleksIS data, required
[database]
host = "localhost"
name = "aleksis"
username = "aleksis"
password = "aleksis"

# Maintenance mode and debug, optional
[maintenance]
debug = true

# Authentication via LDAP, optional
[ldap]
uri = "ldaps://ldap.myschool.edu"
bind = { dn = "cn=reader,dc=myschool,dc=edu", password = "secret" }
map = { first_name = "givenName", last_name = "sn", email = "mail" }

[ldap.users]
search = { base = "ou=people,dc=myschool,dc=edu", filter = "(uid=%(user)s)
→" }

[ldap.groups]
search = { base = "ou=groups,dc=myschool,dc=edu" }
type = "groupOfNames"
# Users in group "admins" are superusers
flags = { is_superuser = "cn=admins,ou=groups,dc=myschool,dc=edu" }

# Search index, optional
[search]

```

(continues on next page)

(continued from previous page)

```
backend = "whoosh"  
index = "/srv/www/aleksis/data/whoosh_index"
```

Configuration in frontend

Everything that does not have to be configured before the AleksIS instance fully starts can be configured in frontend, such as site title and logo.

You can find the configuration options in your AleksIS instance under Admin → Configuration.

3.1.5 Storage

Data in AleksIS is mainly separated into two kinds of data:

- Structured data in a relational database
- Media files in a file storage

Both need to be carefully configured.

Database

The only supported database system in AleksIS is PostgreSQL. Its requirements and basic installation are laid out in the installation chapter.

In order to gain good performance using PostgreSQL, special care should be taken both in configuring the PostgreSQL server itself and AleksIS.

Configuring and tuning PostgreSQL

For help on configuring a performant PostgreSQL database, please refer to the PostgreSQL documentation and related resources. In order to optimise for AleksIS' workload, some helpful hints to consider are:

- AleksIS is a read-heavy workload. Hence, optimise for sufficient `shared_buffers` to hold most of AleksIS' data of the current school term.
- Some modules do extensive calculations in the database, e.g. resolve recurrence rules in the calendar system. This is especially true when using the class register or comparable apps. Ensure that `work_mem` is set high enough – when using the class register in an average high school, it can be advisable to set it as high as 128MB to prevent PostgreSQL from creating temp files
- Failing fast is better than letting users wait indefinitely. Hence, set a reasonable `statement_timeout`, e.g. 5s.
- Using external connection poolers like pgPool or pgBouncer is unnecessary and has no effects apart from reducing performance and security

Settings for AleksIS' database usage

On top of the settings explained in the installation instructions, AleksIS' PostgreSQL usage can be tuned by providing any `OPTIONS` supported by `psycopg`. Notably, these include [\[ConnectionPool\]](#).

Ideally, your connection pool should maintain a minimum of connections to be ready for the usual load of requests during a school day, and allow enough spare connections for load peaks (e.g. during breaks or when new timetables are published).

For an average high school with 1000 students and 100 teachers, we can estimate a *usage factor*. While younger students will mostly check their substitutions when at home, or have this done by their parents, older students might regularly access AleksIS on their smartphones all day long and especially during breaks. Hence, we optimise for 500 users accessing the platform simultaneously. As most requests take less than 100 ms in the database, and accesses don't occur strictly simultaneously, if system resources allow for it, 50 is a good measure for the minimum connection pool size, with a maximum of 100 for peaks (or even 150, if a dedicated PostgreSQL server is used).

Additionally, user interactions should *fail fast* instead of waiting indefinitely. Hence, if the connection pool is exhausted, users should see a timeout error quickly. They will certainly complain, but this is more actionable than having users hammer their touch screens impatiently.

This would result in a configuration like:

```
[database.options.pool]
min_size = 50
max_size = 150
timeout = 5
```

Note

Make sure to also set PostgreSQL's `max_connections` high enough.

File storage

AleksIS needs a writable storage, both for media files (pictures, generated PDF files, and the like), and to store generated frontend assets like the themed CSS stylesheet.

Note

Everything except this media storage can be mounted and used entirely read-only, i.e. to keep the AleksIS installation immutable.

Local filesystem storage

By default, the media storage resides in the local filesystem, in the location defined in the `static.root` configuration key.

Warning

Do not expose the media storage directly through a webserver. AleksIS uses a specially protected storage framework that employs cryptographic tokens to protect user data from URL guessing.

Amazon S3 (or other S3-compatible storage)

AleksIS allows you to configure an Amazon S3 endpoint for media files. This is useful e.g. for loadbalancing with multiple AleksIS instances.

Note

For some background jobs, AlekSIS stores HTML snippets in the media storage for later use. You must ensure your S3 endpoint is part of your Access-Control-Allow-Origin CORS header, so HTML loaded from there can load resources from the AlekSIS instance.

If you want to use an S3 endpoint to store files you have to configure the endpoint in your configuration file (*/etc/aleksis/aleksis.toml*):

```
# Default values
[storage.s3]
enabled = true
endpoint_url = "https://minio.example.com"
bucket_name = "aleksis-test"
access_key_id = "XXXXXXXXXXXXXXXX"
secret_key = "XXXXXXXXXXXXXXXXXXXXXXX"
```

3.1.6 Mail

AlekSIS needs to send mails e.g. for account confirmations, feedback or error reports.

Configure mailing

The mailserver can be configured via the configuration file

```
[mail.server]
host = "mail.example.com"
tls = false
ssl = true
port = 25
user = "mailuser"
password = "password"
```

Name and address for mails sent by AlekSIS can be configured in the webinterface. To configure, visit *Admin* → *Configuration* and click on the *Mail* tab.

Configure mail recipients

You can configure admin contacts in your configuration file, located at */etc/aleksis/*.

```
[contact]
admins = [{"AlekSIS - Admins", "root@example.com"}, {"AlekSIS - Admins2",
→ "root2@example.com"}]
from = 'aleksis@example.com'
```

3.1.7 Authenticating against LDAP

AlekSIS can authenticate users against an LDAP directory (like OpenLDAP or Active Directory). The AlekSIS core can only authenticate and synchronise authenticated users to AlekSIS' database. There are apps that help with tasks like mass-importing accounts and linking accounts to persons in the AlekSIS system (see below).

Installing packages for LDAP support

Installing the necessary libraries for LDAP support unfortunately is not very straightforward under all circumstances. On Debian, install these packages:

```
sudo apt install python3-ldap libldap2-dev libssl-dev libsasl2-dev python3-  
↳ dev
```

Configuration of LDAP support

Configuration is done under the `ldap` section in AleksIS' configuration file. For example, add something like the following to your configuration (normally in `/etc/aleksis`; you can either append to an existing file or add a new one):

```
[ldap]
uri = "ldaps://ldap.myschool.edu"
bind = { dn = "cn=reader,dc=myschool,dc=edu", password = "secret" }

[ldap.users]
search = { base = "ou=people,dc=myschool,dc=edu", filter = "(uid=%(user)s)"
↳ " }
map = { first_name = "givenName", last_name = "sn", email = "mail" }

[ldap.groups]
search = { base = "ou=groups,dc=myschool,dc=edu" }
type = "groupOfNames"
# Users in group "admins" are superusers
flags = { is_superuser = "cn=admins,ou=groups,dc=myschool,dc=edu" }
```

3.1.8 Registration and user invitations

In addition to central management of user accounts, AleksIS allows self-registration by users. Registration can be either fully open, or based on personal invitations.

In a system handling critical data, access control should be as tight as possible. However, there are scenarios where central account creation is not feasible, e.g. for optional guardian accounts. In such a scenario, the invitation system allows for processes like handing out invitation codes as a letter or through e-mail campaigns.

Configuration

Registration

Registration can be enabled via the configuration interface (Administration → Configuration) in the frontend.

In the Authentication tab, click the checkbox `Enable signup` to enable signup for everyone. A menu item will be added for public registration.

Warning

Do not enable this feature unless you intend to run a public AleksIS instance.

Before enabling registration, you should consider restricting allowed usernames. By default, all ASCII characters are allowed in usernames. Often, it is advisable to not allow special characters. This often depends on the systems that will be linked to AlekSIS.

To restrict usernames to a certain format, a regular expression can be defined in the Regular expression for allowed usernames preference. For example, to restrict the username to lower case letters and numbers, and beginning with a number, the regex can be set to `^[a-z][a-z0-9]+$`.

User invitations

In the same location as public registration, the invitation system can be enabled.

- Authentication
 - Enable invitations: Click to enable invitations.
 - Length of invite code: Length of invitation code packets, defaults to 5.
 - Size of packets: Configure how many packets are generated, defaults to 3.

By default, an invitation code looks like the following: abcde-abcde-abcde.

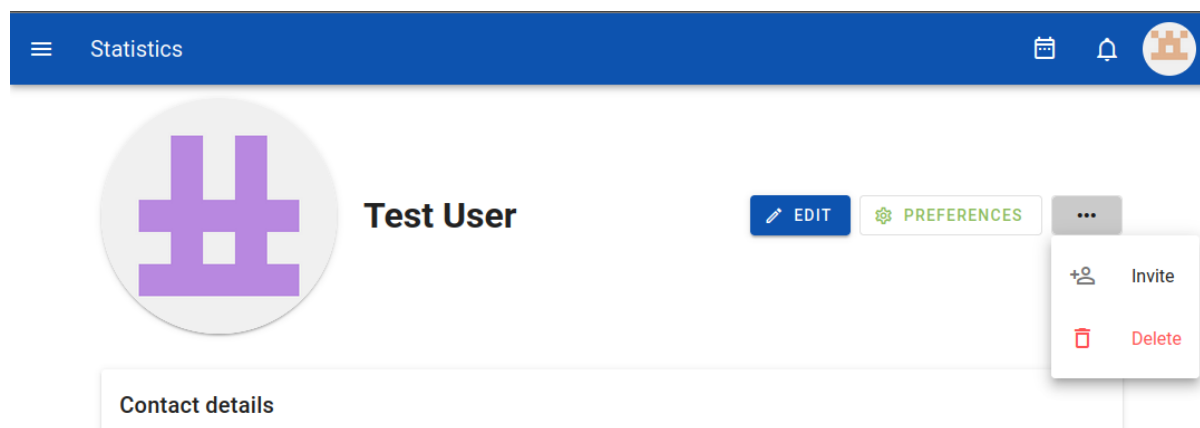
A menu item will become available for users to enter their invitation code.

Usage

Invite existing person

To invite an existing person, open the person in AlekSIS and click the Invite user menu item.

The invitation will be sent to the person's email address, and can only be used by this person. Upon registration, the new account will automatically be linked to the existing person.



Note

Before using this feature, make sure to read and understand [The concept of persons](#) (page 39).

3.1.9 Social accounts

AlekSIS can authenticate users against third party applications using OAuth2 or OpenID.

This can be used to grant access to persons whose credentials shall not be managed in AlekSIS itself, for example because another authentication provider is already used throughout the school,

or for guardians that can or should for some reason not get a local account, or similar situations.

Warning

Social accounts are **not** working with two factor authentication! If a user authenticates with a social account, the two factor authentication is ignored on login (but enforced for views that require two factor authentication later).

Configuring social account provider

For available providers, see documentation of [django-allauth](#)⁸.

A new social account provider can be configured in your configuration file (located in `/etc/aleksis/`).

Configuration examples:

```
# GitLab
[[auth.providers.gitlab.APPS]]
client_id = "<client_id>"
secret = "<client_secret>"
settings = { gitlab_url = "https://gitlab.example.com" }

# Generic OpenID Connect
[[auth.providers.openid_connect.APPS]]
client_id = '<client_id>'
secret = '<client_secret>'
name = 'Service Name'
provider_id = 'service_name'
settings = { server_url = 'https://example.org' }
```

After configuring a new authentication provider, you have to restart AleksIS.

Match local accounts to social accounts by username

You can configure AleksIS to automatically match local accounts to social accounts by their username. To do this, set the following configuration:

```
[auth]
socialaccount_username_matching = true
```

Warning

Only activate this behavior, if you are completely sure that you want to match local accounts to social accounts by their username and that the third-party provider can be trusted.

⁸ <https://docs.allauth.org/en/latest/socialaccount/providers/index.html>

3.1.10 Monitoring and health checks

Configuration

Thresholds

Thresholds for health checks can be configured via config file (/etc/aleksis).

```
[health]
disk_usage_max_percent = 90
memory_min_mb = 500

[backup.database]
check_seconds = 7200

[backup.media]
check_seconds = 7200
```

Status page

AlekSIS' status page shows information about the health of your AlekSIS instance. You can visit it via the left navigation bar (Admin → Status).

The page shows information about debug and maintenance mode, a summary of your health checks and the last exit status of your celery tasks. This page can not be used as a health check, it will always return HTTP 200 if the site is reachable.

Health check

The health check can be used to verify the health of your AlekSIS instance. You can access it via the browser (<https://aleksis.example.com/health/>) and it will show you a summary of your health checks. If something is wrong it will return HTTP 500.

It is also possible to get a JSON response from the health check, for example via curl. You only have to pass a valid Accept: application/json header to your request.

The health check can also be executed via aleksis-admin:

```
$ aleksis-admin health_check
```

Monitoring with Icinga2

As already mentioned, there is a JSON endpoint at <https://aleksis.example.com/health/>. You can use an json check plugin to check separate health checks or just use a HTTP check to check if the site returns HTTP 200.

Performance monitoring with Prometheus

AlekSIS provides a Prometheus exporter. The exporter provides metrics about responses and requests, e.g. about response codes, request latency and requests per view. It also provides data about database operations.

The metrics endpoint can be found at <https://aleksis.example.com/metrics>. In the default configuration it can be scraped from everywhere. You might want to add some webserver configuration to restrict access to this url.

To get metrics of your AleksIS instance, just add the following to `prometheus.yml`

```
- job_name: aleksis
  static_configs:
    - targets: ['aleksis.example.com']
  metrics_path: /metrics
```

Rules for prometheus alertmanager

If you are using the prometheus alertmanager, it is possible to create some alerting rules so that an alert is fired when your AleksIS instance is slow or something.

```
groups:
- name: aleksis
  rules:
  - alert: HighRequestLatency
    expr: histogram_quantile(0.999, sum(rate(django_http_requests_latency_
    ↳seconds_by_view_method_bucket{instance="YOUR-INSTANCE", view!~"prometheus-
    ↳django-metrics|healthcheck"}[15m])) by (job, le)) < 30
    for: 15m
    labels:
      severity: page
    annotations:
      summary: High request latency for 15 minutes
```

Grafana dashboard

There is a Grafana dashboard available to visualise the metrics.

The dashboard is available at <https://grafana.com/grafana/dashboards/9528>.

3.1.11 Background tasks

Operations that are expected to take a long time are run as background tasks. For this, at least one [Celery](https://celeryproject.org/)⁹ worker has to be running, e.g. by coupling it with uWSGI as laid out in [Configure uWSGI](#) (page 45).

If a task is triggered from the AleksIS frontend, for example by starting an import job, a progress page is displayed, and the result of the job is waited for. When the page is closed while the job has still not finished, an information bar showing the progress will be visible until it has finished.

Periodic tasks

Some tasks are also run on a schedule. For example, the backup job is run on a regular basis.

All tasks in AleksIS that are expected to run have a default schedule, which is registered when migrating the database. Changing this default schedule is currently only possible through the Django Admin backend, under *Admin* → *Backend Admin*.

Under the *Periodic Tasks* app, you can define schedules and tasks. The names of tasks you can add manually are documented in the respective sections of the manual.

⁹ <https://celeryproject.org/>

3.1.12 Data checks

Data checks are AlekSIS' mechanism for highlighting issues with the contents of the database. These checks are not of a technical nature, but strictly concern the contextual integrity of the data stored.

Verify data checks

In the menu under Administration → Data checks, the status of all known checks can be verified.

The first card shows the current global check state. If any data checks reported issues, they will be listed here. In that case, administrators can choose between options provided by the data checks to resolve the issues.

Note

Details about the checks and solve options are described in the respective chapters of the manual.

Configure notifications

In the General tab of the configuration interface, you can configure email notifications for problems detected by the data checks.

- General
 - Send emails if data checks detect problems: Enable email notifications
 - Email recipients for data checks problem emails: Choose recipient persons
 - Email recipient groups for data checks problem emails: Choose recipient groups

Data checks normally run once per hour, and if notifications are enabled, results will be mailed to the selected recipients if problems are detected.

3.1.13 Providing important information to users using the dashboard

The dashboard is a central place for providing important information to users. This is done by so-called dashboard widgets provided by the Core and apps.

Built-in dashboard widgets

External link widget

The external link widget will show a link to an external site on the dashboard, optionally with an icon or picture next to it. It therefore provides the following additional attributes:

- **URL:** The URL of the external site.
- **Icon URL:** The URL of the icon or picture shown next to the link.

As link title, the widget title will be used.

Static content widget

The static content widget allows to display custom static information on the dashboard, It therefore provides the following additional attribute:

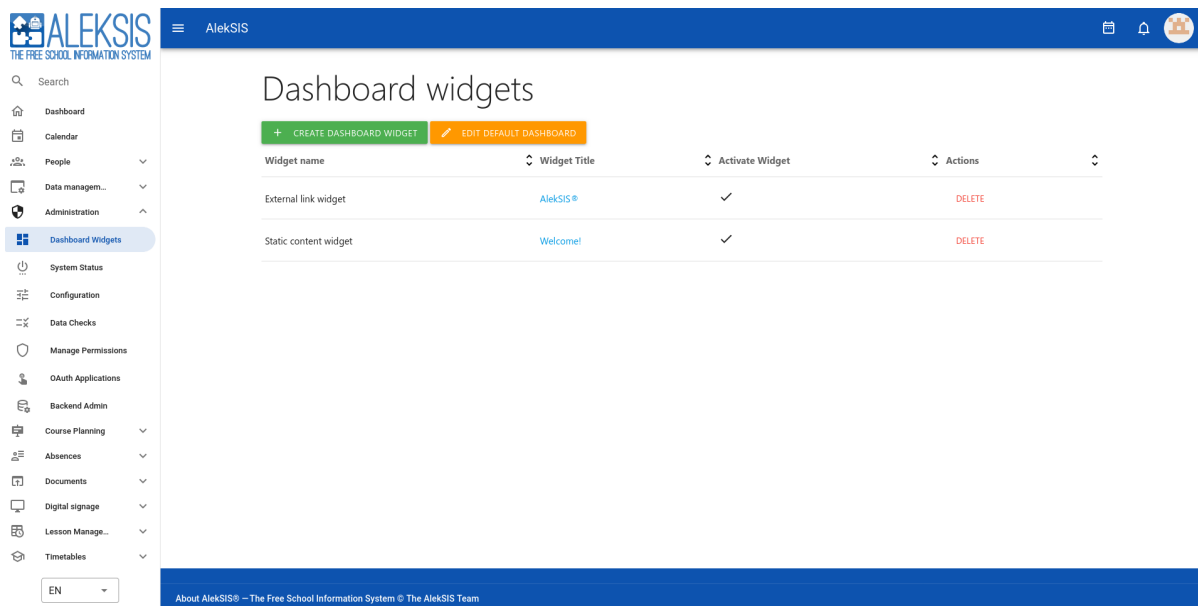
- **Content:** The content of the widget. HTML can be used for formatting.

More dashboard widgets from apps

In addition to the built-in widgets, apps can provide their own dashboard widgets. Best examples for such apps are currently *AleksIS-App-DashboardFeeds* and *AleksIS-App-Chronos*.

Add and configure dashboard widgets

If you want to add a new dashboard widget, you can do so by adding the dashboard widget at *Admin* → *Dashboard widgets*. There you will see all currently configured dashboard widgets and can add new ones using the *Create dashboard widget* button which will ask your for the widget type.



Each dashboard widget has at least the following attributes

- **Widget Title:** The title of the widget (will be shown in some widgets).
- **Activate Widget:** If this isn't checked, the widget will not be shown.
- **Widget is broken:** If this is checked, the widget will be shown but the user will get a message that this widget is currently out of order because of an error. This shouldn't be checked by yourself, but might be activated automatically by a widget if it encounters an error. If this case enters, you should check for the cause of the error and fix it. After that, you can unmark the widget as broken.
- **Size on different screens:** The size of the widget on different screens. We work with a grid system containing a maximum of 12 columns. So, one column is 1/12 of the screen width. The width in the following fields has to be entered as number of columns (1 to 12).
 - **Size on mobile devices:** The size of the widget on mobile devices (600px and less).
 - **Size on tablet devices:** The size of the widget on desktop devices (600px - 992px).
 - **Size on desktop devices:** The size of the widget on desktop devices (992px - 1200px).
 - **Size on large desktop devices:** The size of the widget on large desktop devices (1200px and above).

All other attributes are specific to the widget type and are explained in the documentation of the widget.

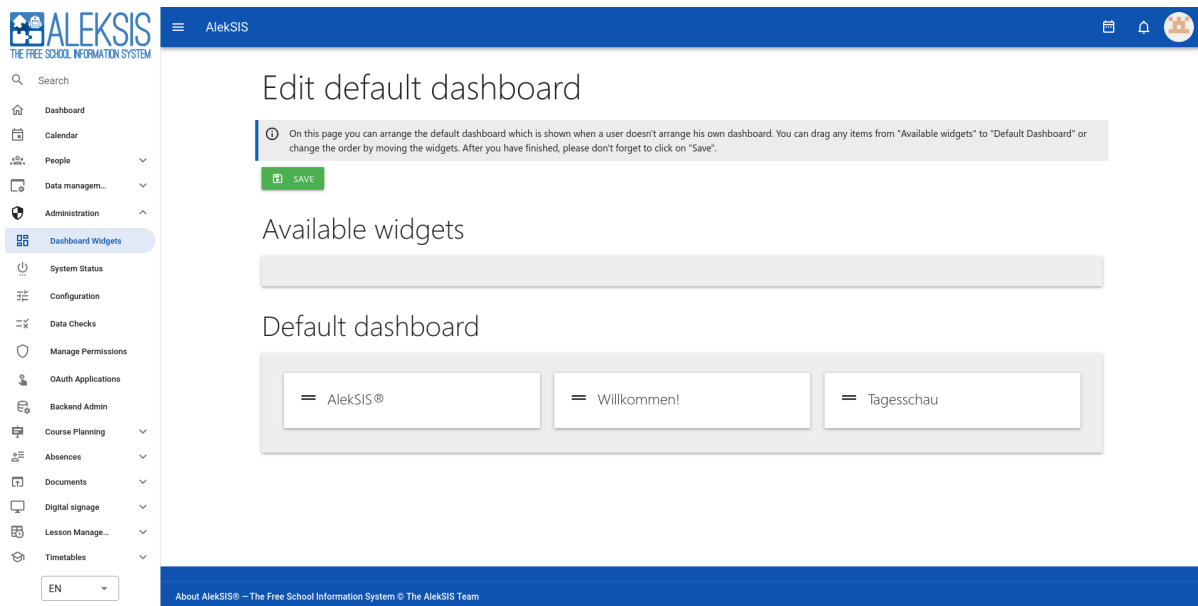
The screenshot shows the 'Create Icalendar Widget' form in the AleksIS interface. The form has a blue header with the AleksIS logo and navigation icons. A sidebar on the left lists various system functions. The main form area contains the following fields:

- Widget Title:** A text input field.
- Activate Widget:** A checkbox.
- Widget is broken:** A checkbox.
- Size on mobile devices:** A numeric input field with the value '12'. Below it, text indicates '<= 600 px, 12 columns'.
- Size on tablet devices:** A numeric input field with the value '12'. Below it, text indicates 'Size on tablet devices'.
- Size on desktop devices:** A numeric input field with the value '6'. Below it, text indicates '> 600 px, 12 columns'.
- Size on large desktop devices:** A numeric input field with the value '4'. Below it, text indicates '> 992 px, 12 columns'.
- ICalendar URL:** A text input field.
- Base URL of related calendar:** A text input field.
- Number of displayed events:** A numeric input field with the value '5'. Below it, text indicates 'The widget will have a link to visit a related website to see more events'.

Setup a default dashboard

To make the configured dashboard widgets accessible to all users, we recommend to configure the default dashboard. If you don't do so, the dashboard widgets will only be available to users if they customise their dashboard.

The default dashboard can be configured via *Admin* → *Dashboard widgets* → *Edit default dashboard*. The edit page works exactly as the page described in [Customising the dashboard](#) (page 13).



Preferences

The behavior of the dashboard can be configured via *Admin* → *Configuration* → *General*. The following settings are available:

- **Show dashboard to users without login:** If this is checked, the dashboard will be also shown to users who are not logged in.

Warning

That won't work with all dashboard widgets. Some widgets, like the timetable widgets, require a logged in user.

- **Allow users to edit their dashboard:** With this preference, system administrators can decide whether users can edit their own dashboard as described in [Customising the dashboard](#) (page 13).
- **Automatically update the dashboard and its widgets sitewide:** If enabled, the dashboard will be updated automatically every 15 seconds.

3.2 Managing timetable data and substitution plans in Chronos

3.2.1 Managing timetable and substitution data

Creating timetables

For interactively planning timetables using time grids, please use AleksIS-App-Lesrooster.

Alternatively, data can be imported from an external source. The official distribution handbook contains documentation about the existing integration apps.

Managing substitutions

You can use *Timetables* → *Plan substitutions* to interactively plan substitutions and cancellations for lessons affected by absences. This consists of the following workflow:

1. Create absences of teachers at *Absences* → *Absences* (a link leading to this page is also including in the substitution planning frontend).
2. See all affected lessons at *Plan substitutions* and assign new teachers, subjects, and/or rooms. It is also possible to cancel the lesson. Students and teachers then can see the changed data and also will be notified, if activated.

3.2.2 Setup notifications about current changes

Users can get notifications about current changes to their personal timetables. To activate this behavior, the system administrator has to ensure multiple things:

- The notifications have been activated in the preferences (see below).
- There is at least one notification channel available to your users (cf. core-admin-notifications).

Preferences

You can customize the way how and when notifications are sent at the configuration page at *Admin* → *Configuration* → *Timetables*:

- **Send notifications to affected teachers when substitution lessons are created or edited:** With this checkbox, the whole feature can be activated or deactivated.
- **Trigger mode for lesson event alarms:** Notifications can be sent at a fixed time one or more days earlier or relative to the events start time.
- **How many days in advance should lesson event alarms be sent?** Here the number of days can be configured notifications will be sent before the actual affected day. A common value is one or two days.
- **Time for sending lesson event alarms:** At this time, the notifications for the next days will be sent (if fixed select, see above). This is only used if the changes are created before the period configured with the above mentioned option. If they affect a day in this period, the notification will be sent immediately.
- **How much in advance should lesson event alarms be sent?:** This says how early notifications should be sent (in relation to the lesson start time).

3.2.3 System-wide settings for timetable and substitution display

The display of timetables and substitution plans can be customised under *Admin* → *Configuration* → *Timetables*. The section contains the following preferences:

- **Use parent groups in timetable views:** If a lesson or substitution has only one group and this group has parent groups, these groups will be shown.
- **Relevant days for substitution plans:** Substitution plans are only created for these days.
- **Time when substitution plans switch to next day:** At this time, the generated substitution plan will switch to the next day.
- **Number of days shown on substitutions print view:** Here you can set the number of days that will be shown on the substitutions print view (PDF file).

- **Show header box in substitution views:** The header box shows all affected/absent teachers/groups for the current day.
- **Show parent groups in header box in substitution views instead of original groups:** This setting determines the same setting as *Use parent groups in timetable views*, but for the printout.
- **Group types to show in timetables:** Only groups with these types will be shown in the timetable select.
- **Lesson calendar feed color:** Default color for lesson calendar feed.
- **Supervision calendar feed color:** Default color for supervision calendar feed.
- **Days of the week that appear in the timetable:** Only these days are shown in the timetable calendar.

3.3 Setting up a digital class register with AleksIS-App-Alsijil

3.3.1 Defining base data

With sufficient authorisation, two additional menu items appear in the class register menu.

Absence reasons

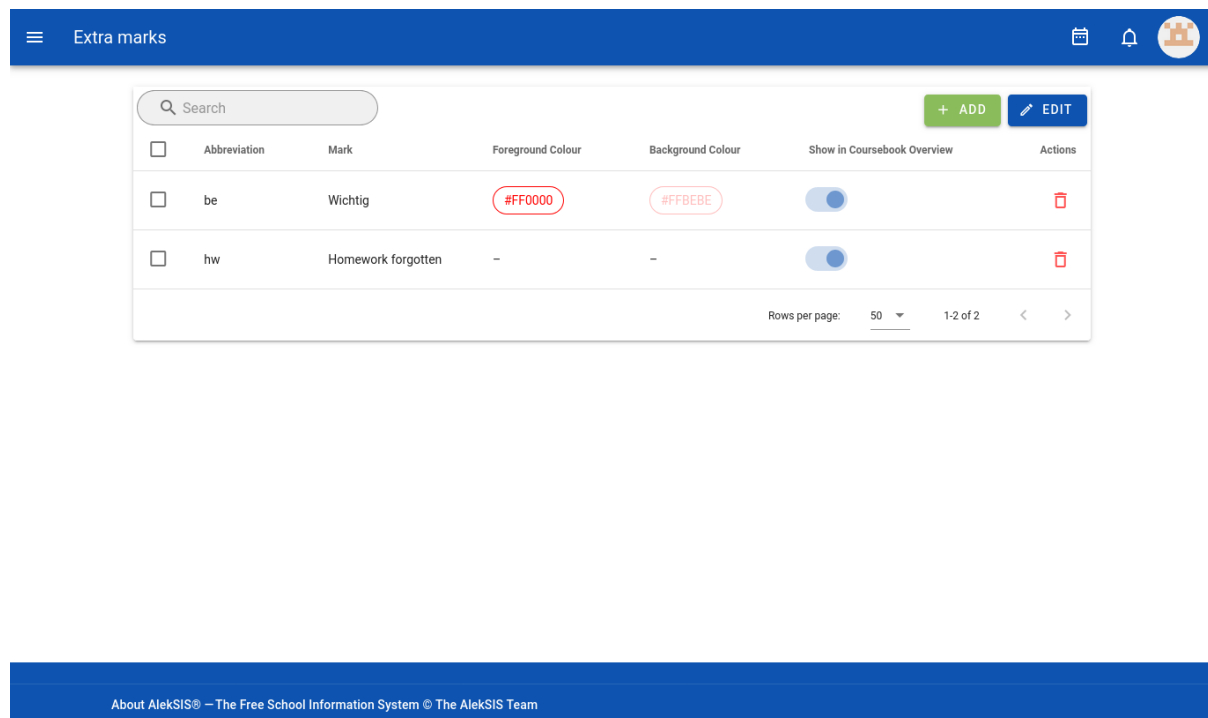
Students can be absent due to different reasons. These absence reasons can be managed at *Absences* → *Absence Reasons*. Please see the Kolego documentation for further information. To show them in the class register, set the tag *Class Register*.

Short name	Name	Colour	Count as absent	Default Absence Reason	Tags
a	absent	#AABBCC	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Will disable previous default when enabled	Class Register +
D	Dienstlich	#AABBCC	<input checked="" type="checkbox"/>	<input type="checkbox"/> Will disable previous default when enabled	<input checked="" type="checkbox"/> Class Register +
S	Sonderurlaub	#AABBCC	<input checked="" type="checkbox"/>	<input type="checkbox"/> Will disable previous default when enabled	+
Fortbildung	Fortbildung	#AABBCC	<input checked="" type="checkbox"/>	<input type="checkbox"/> Will disable previous default when enabled	+
kr	krank	#AABBCC	<input checked="" type="checkbox"/>	<input type="checkbox"/> Will disable previous default when enabled	+
?	?	#AABBCC	<input checked="" type="checkbox"/>	<input type="checkbox"/> Will disable previous default when enabled	+
K	Krank	#AABBCC	<input checked="" type="checkbox"/>	<input type="checkbox"/> Will disable previous default when enabled	+
e	excused	#AABBCC	<input checked="" type="checkbox"/>	<input type="checkbox"/> Will disable previous default when enabled	+

Rows per page: 50 1-8 of 8

Extra marks

Some remarks are repeated over and over again, such as 'Homework forgotten'. In order not to have to write this again and again in the remark field, additional marks can be set, which then only have to be clicked on in the class register. To manage them, open *Class register* → *Extra marks*.



3.3.2 System-wide settings for the digital class register

The behaviour of the digital class register can be customised under *Administration* → *Configuration* → *Class Register*. The section contains the following preferences:

- **Set time range for which documentations may be edited:** This can be used to allow editing documentations only for the past or also for the future.
- **User is allowed to register absences for members of groups the user is an owner of with these group types:** If a user is owner of a group with one of those group types, he is allowed to register absences for longer time periods for the persons in this group.
- **User is allowed to view coursebook statistics for members of groups the user is an owner of with these group types:** If a user is owner of a group with one of those group types, he is allowed to see class register statistics for the persons in this group.
- **Group type of groups to be shown first in the group select field on the coursebook overview page:** Groups with this group type will be shown before all other groups in the select on the coursebook overview page (e. g. for classes).

3.4 Seating plan administration

3.4.1 Permissions

Group owners can always view, create, edit and delete seating plans for their groups. Also, they can view plans of parent groups.

The Seating plans menu entry is only displayed, when at least one seating plan can be viewed by a person.

To give persons who are not group owners permissions, use the following:

- **stoelindeling.view_seatingplan:** Global or object permission to view all seating plans or a specific one

- **stoelindeling.add_seatingplan**: Global permission to create seating plans
- **stoelindeling.change_seatingplan**: Global or object permission to edit all seating plans or a specific one
- **stoelindeling.delete_seatingplan**: Global or object permission to delete all seating plans or a specific one

Persons with the global permission **stoelindeling.add_seatingplan** are also allowed to view the list of all seating plans.

3.5 Providing time-based documents with Resint

3.5.1 Uploadable posters for time-based documents

Posters are documents that can be manually supplied in a time-based manner. Example use cases are canteen menus, that are provided weekly by an external supplier, who will only get privileges to upload this poster as a PDF file.

Poster files can be uploaded for a defined time period, and AleksIS will then deliver the currently valid version under a stable menu item and URL.

Defining poster groups

Uploadable posters are categorised into poster groups, with each poster group representing one time-based document that AleksIS will provide. Each poster group has:

- A name
- A URL slug making up the final, stable URL
- A schedule defining the cycle for providing the document
- A default PDF file for periods for which no document is uploaded

Permissions to upload documents are managed per poster group as well.

Creating poster groups

Poster groups are created from the menu under the *Documents* → *Poster groups* menu item. After clicking the *Create new poster group* button or the *Edit* action for any poster group, the following form allows editing the details of the poster group.

In addition to the properties explained above, the form provides two options to configure if and how the document will be presented to users. It is possible to include or exclude the document from the main menu, and to allow access by anonymous users. Documents that are hidden from the menu can be used if they should only be accessible through links from external sites, or loaded by a digital signage system.

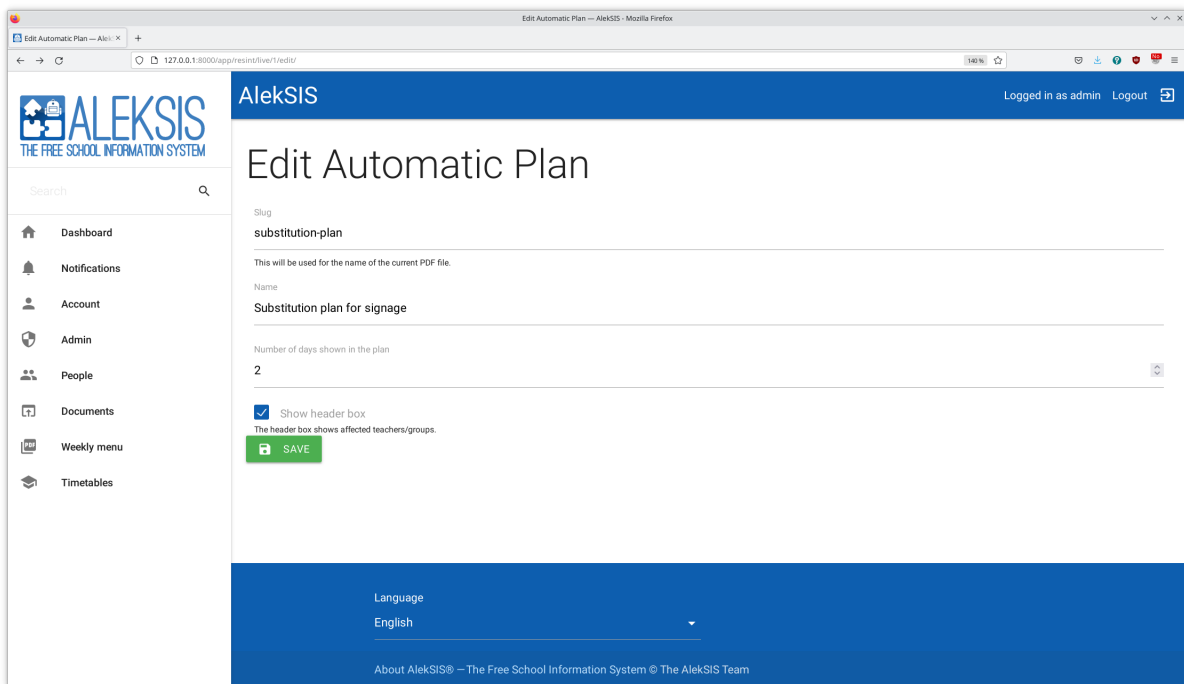
The stable URL for the poster group can be copied from the link in the *Filename* column of the *Poster groups* list.

3.5.2 Configuring live documents

Live documents are another type of time-based documents. In contrast to posters, they are not uploaded manually, but automatically updated by some trigger.

Types of live documents are provided by other AlekSIS apps. All live document types provided by apps can then be configured from the *Documents* menu.

An example for a live document is an automatically generated substitution plan, which can be linked to from a website, or displayed on digital signage.



All fields except the slug are provided by the app. The slug again makes up the stable URL the live document will be provided under, which can be copied from the live document list.

The app providing a live document type will take care of updating the document. A live substitution plan might thus be re-generated whenever timetable or substitution data changes.

3.6 Importing timetables and substitutions from Untis

3.6.1 Untis data and their relation to AleksIS

Untis is a proprietary timetable management software which is popular in the German-speaking area, but used internationally. AleksIS provides functionality to automatically import data from Untis into data models of the core, *Cursus*, the app for course planning, *Lesrooster*, the timetable planning app, and *Chronos*, the timetable and substitution app.

Currently, we only support importing data from the MySQL database of [Untis MultiUser](https://www.untis.at/produkte/untis-das-grundpaket/multiuser)¹⁰. This is done through configurable background tasks which are executed in a specific interval or at specific points in time.

Supported Untis features

Not all features of Untis are supported in AleksIS. The following information from Untis can be imported into AleksIS:

- Terms
- Holidays
- Classes, teachers, subjects
- Rooms, supervision areas (corridors in Untis)
- Lesson and break times

¹⁰ <https://www.untis.at/produkte/untis-das-grundpaket/multiuser>

- Timetable data (lessons, supervisions)
- Absences, absence reasons
- Substitutions, extra lessons, cancellations
- Events
- Exams

The Untis integration supports the versioning features of Untis. By default, the most recent version of each object is imported.

Currently, the following features are known not to be supported:

- Students, student groups, student choices
- Prebookings
- Statistical data
- Special rooms (subject and group rooms)

AlekSIS does not support so-called “day texts” from Untis. These are incompatible with AlekSIS’ announcement feature, which can be used as a replacement.

3.6.2 Setting up the Untis integration

Requirements

To use the importer, you must have a current Untis MultiUser license (version 2019 and above) and a MySQL database which is reachable by the AlekSIS server. How to configure Untis to use this database is described in the [Untis MultiUser Manual¹¹](#).

In addition to the technical [Prerequisites](#) (page 42) of the AlekSIS core itself, a few extra system packages are required:

```
apt install libmariadb-dev
```

The MySQL (or [MariaDB¹²](#)) server must be reachable from the AlekSIS server, and a user account in the database is needed. It is sufficient to create this user with SELECT permissions. On the MySQL shell, you can create such a user with something like:

```
CREATE USER `aleksis`@`aleksisserver` IDENTIFIED BY 'securepassword';
GRANT SELECT ON `untis`.* TO `aleksis`@`aleksisserver`;
FLUSH PRIVILEGES;
```

Configure database connection

In the AlekSIS configuration file (cf. [Configuration files and format](#) (page 48)), you have to set the following settings:

```
[untis.database]
enabled = true
name = "untis"
user = "aleksis"
```

(continues on next page)

¹¹ https://help.Untis.at/hc/de/article_attachments/360004504079/Untis_MultiUser.pdf

¹² <https://mariadb.org>

(continued from previous page)

```
password = "securepassword"
host = "mysqlserver"
port = 3306
```

Preferences

The preferences for the import can be set from the menu under *Admin* → *Configuration* → *Untis*.

Configure the school ID

The only required preference is the Untis school ID. You need to provide this in all cases, even if your Untis database hosts only one school.

Warning

If your Untis database hosts several schools, but you forget to configure the school ID, data corruption may occur!

Customise how data are imported

The behaviour of the import can be customised in several ways. The following preferences are available:

- **Update values of existing subjects:** This will update the values of already existing subjects if Untis has different data.
- **Update short name of existing persons:** This will update the short name of already existing persons if Untis has different data.
- **Update name of existing persons:** This will update the name of already existing persons if Untis has different data.
- **Update short name of existing groups:** This will update the short name of already existing groups if Untis has different data.
- **Update name of existing groups:** This will update the name of already existing groups if Untis has different data.
- **Overwrite group owners:** This will update the group owners of already existing groups if Untis has different data.
- **Group type for class groups:** This group type will be set for all class groups.
- **Update name of existing rooms:** This will update the name of already existing rooms if Untis has different data.
- **Update existing supervision areas:** This will update the values of already existing supervision areas if Untis has different data.
- **Use course groups:** This will search course groups (groups for each subject in a class) instead of setting the classes as groups.
- **Create non-existing course groups:** In combination with `_Use course groups_` being enabled, this will create new course groups if no matching group was found.
- **Group type for course groups:** This group type will be set for all course groups.

- **Register a data problem if a course group has been not found:** When this is activated, the import will register a data problem if no matching course group was found, independent of whether a new course group was created or not.
- **Ignore incomplete substitutions:** If this is activated, Untis won't import any substitutions which are not cancelled or without a new room, new teacher or new subject .

Scheduling import tasks

The integration provides different [Periodic tasks](#) (page 57) to import the data from Untis:

- `untis_import_mysql_current_term`: This will import all data from the **current** Untis term.
- `untis_import_mysql_future_terms`: This will import all data from all **future** Untis terms, but not from the current.
- `untis_import_mysql_all_terms`: This will import all data from **all** Untis terms which are in the database.
- `untis_import_mysql_current_next_term`: This will import all data from the **current and the directly following** Untis term.
- `untis_import_mysql_current_future_terms`: This will import all data from the **current and all future** Untis terms.

We suggest using `untis_import_mysql_current_next_term` as a task because this will ensure that all current data are up-to-date, but also that the next timetable version is also already imported when it becomes relevant.

In general, all tasks will do nothing if there is no matching Untis term.

To use these tasks, you have to add them as periodic tasks. By default, they will import the most recent plan version from Untis. To select a specific version (i.e. to import an older snapshot), you can pass the `version` argument in the tasks.

How existing data is matched

If there are already existing data in AlekSIS' database, the Untis import will always try to combine these data. The main data field used for this is the `short_name` field (cf. [The concept of groups](#) (page 40)). If the data were imported one time, each object in Chronos will save the respective ID from Untis to make sure that the data are properly updated at the next import.

The import is thus idempotent.

3.7 Importing data from CSV-like sources

3.7.1 Base concept of the generic importer

For importing comma-separated or, more generally, any sort of column-based data in text files, AlekSIS provides a generic importer.

The importer works with so-called *import templates*. These templates set a field type for each column in a file. With this information, the importer will know how to interpret each cell.

Default templates

For import sources that have already been tested, AleksIS provides import templates that are readily available.

Right now, the following import sources are supported:

- [Pedasos¹³](#), a school management solution popular in Northern Germany
- [Schild-NRW¹⁴](#), the obligatory school management solution in North-Rhine Westphalia, Germany
- [Untis¹⁵](#), a proprietary timetable management software (see [Importing data from Untis](#) (page 72))

These software products either provide hard-coded export mechanisms, or allow the creation of report/export templates to generate CSV-like output.

Custom templates

In addition to the integrated default templates, custom templates can be created by administrators. Creating custom import templates currently requires hand-crafting YAML files that can then be uploaded through the user interface.

However, this feature is currently deliberately undocumented.

3.7.2 Configuring the import

Setting preferences

There are some preferences that should be set before starting the first import to ensure that everything works correctly. Under *Admin* → *Configuration* → *CSV import*, all import templates can be configured.

- **Group type for department groups:** This group type will be set for department (i.e. subject) groups.
- **Prefix for long names of department groups:** This prefix will be added to the long name of all department group names.
- **Languages for date parsing:** Here you can set languages you want to use for parsing dates from the CSV file. Takes comma-separated data without spaces (e. g. *de,en_US*).

In addition, you should set the **country code for phone number parsing** under the *Internationalisation* tab correctly.

3.7.3 Exporting data from various systems

Pedasos

Pedasos is a school management software mainly used in the North of Germany (Schleswig-Holstein, Hamburg).

Please pay attention to the order of the imports: You should start with the classes and end with the guardians.

1. Pedasos: Classes

¹³ <https://ostertun.de/produkt.html>

¹⁴ <https://www.svws.nrw.de/download/schild-nrw>

¹⁵ <https://untis.at>

2. Pedasos: Courses
3. Pedasos: Teachers
4. Pedasos: Students
5. Pedasos: Guardians 1
6. Pedasos: Guardians 2

To use these templates, you have to configure so-called *lists* in Pedasos instance. AlekSIS provides example list definitions to export CSV data matching the import templates.

3.7.4 Importing data

You can find the import form under *Admin* → *Data management* → *CSV import*. Please be sure to select a matching school term (especially when importing groups). Then select the file you want to import and a matching import template.

Warning

Please double-check if the combination of file, template, and school term is correct; otherwise, existing data may be corrupted!

To start the actual import, press the button *Import data*. Now you should see a progress bar with the current import status and all errors that may occur.

Importing multiple CSV files

Multiple CSV files can be uploaded in one batch by putting them in a ZIP archive.

Note

For this to work, the CSV files need to exactly follow the same pattern and be exactly homogeneous.

Accompanying files (photos)

The import of photos is supported. To do this, a CSV with at least the unique import reference and a filename needs to be placed in a ZIP archive, together with the files to be imported.

3.7.5 Importing data from Untis

Untis is a proprietary timetable management software which is popular in the German-speaking area, but used internationally. We provide some import templates for importing basic data from Untis to *Core*, *Cursus*, and *Lesrooster*:

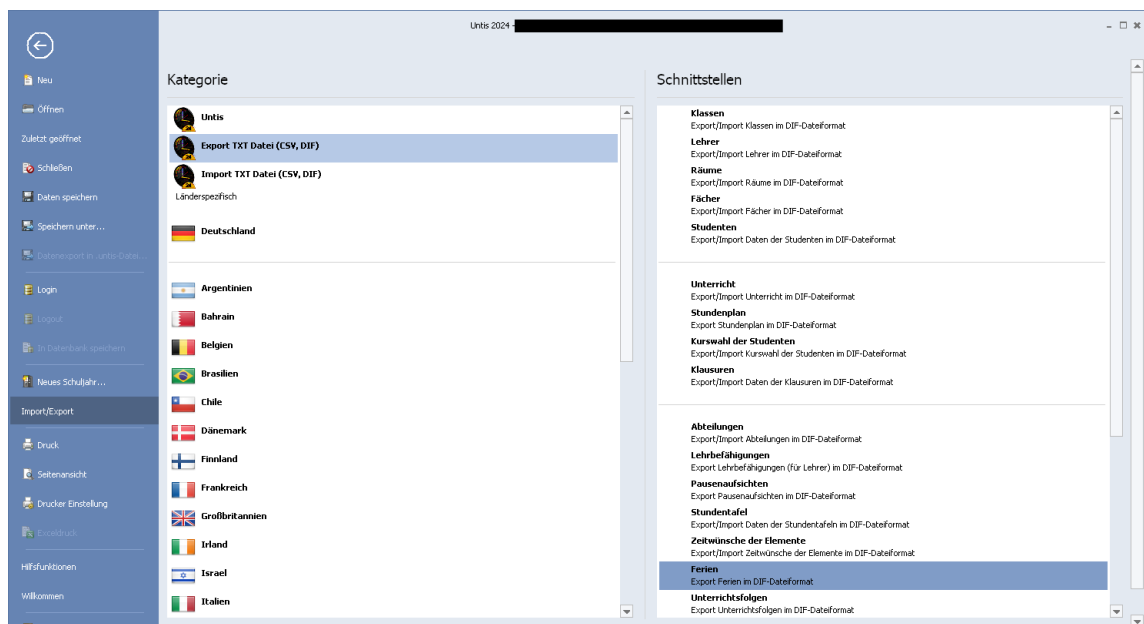
Untis Filename	Name in Untis	App	Model
GPU001.txt	Stundenplan (Timetable)	Lesrooster	Lesson/LessonBundle
GPU002.txt	Unterricht (Courses with quotas)	Cursus	Course/CourseBundle
GPU003.txt	Klassen (Classes)	Core	Group
GPU004.txt	Lehrer (Teachers)	Core	Person
GPU005.txt	Räume (Rooms)	Core	Room
GPU006.txt	Fächer (Subjects)	Cursus	Subject

Warning

To ensure a correct import, all imported objects should have both a short name and a long name, for teachers a first and a last name.

How to export from Untis and import in AleksIS

1. Open Untis
2. Click on *File* in the top navbar
3. Select *Import/Export* and then *Export TXT Datei (CSV,DIF)*



4. Select the file you want to export (see above for options)
5. When asked for export settings, please select *Komma* as *Trennzeichen* and *"* as *Textbegrenzung*. Please ensure that *Encoding: UTF-8* is checked.



To do a full import, you should use the following order:

1. Manually create time grid with slots in AlekSIS (There is no export option for this in Untis.)
2. Import the files in the following order:
 1. GPU005.txt: Räume (Rooms)
 2. GPU006.txt: Fächer (Subjects)
 3. GPU004.txt: Lehrer (Teachers)
 4. GPU003.txt: Klassen (Classes)
 5. GPU002.txt: Unterricht (Courses with quotas)

Then go to *Lesson Planning* → *Validity Ranges* and click the *Import data* button next to the validity range you want to import data for. Use that to import the following file:

6. GPU001.txt: Stundenplan (Timetable)

3.8 Show data from external sources on dashboard

3.8.1 Introduction

AlekSIS provides dashboard widgets for displaying feeds from external resources. The following widgets are normal dashboard widgets and can be added and configured like described in [Add and configure dashboard widgets](#) (page 59).

3.8.2 RSS feed widget

This widget, unlike the name suggests, allows RSS, Atom and JSONFeed feeds to be parsed and displayed on the dashboard. Only the most recent article is displayed, with a short caption and optionally an image. For this, the feed must contain a JPG image as an enclosure (this is not the case by default in every CMS, but most have plugins for this). In addition, the news source is linked on the dashboard.

- **RSS feed source URL:** The URL of the source feed

- **Base URL:** The home or base URL of the news provider
- **Text only:** With this being enabled, no images will be shown.

Note

The RSS widget provides a task to pull data. The task `get_feeds` updates all active RSS feeds inside AleksIS. We recommend to run the task every 5 to 10 minutes. The task is automatically scheduled every 10 minutes; this can be changed as described in [Periodic tasks](#) (page 57).

AleksIS Logged in as admin Logout DIBT

Create Rss Widget

Widget Title
NASA

☒ Activate Widget
☐ Widget is broken
Size on mobile devices
 12

<= 600 px, 12 columns
Size on tablet devices
 12

> 600 px, 12 columns
Size on desktop devices
 6

> 992 px, 12 columns
Size on large desktop devices
 4

> 1200 px, 12 columns

RSS feed source URL
https://www.nasa.gov/rss/dyn/breaking_news.rss

Base URL of related website
<https://www.nasa.gov/>

The widget will have a link to visit a related website to read more news

☐ Text only
 Do not show an image to depict the news item

SAVE

Language
 English

About AleksIS® — The Free School Information System © The AleksIS Team



3.8.3 iCalendar feed widget

This widget displays the upcoming events of an online calendar on the dashboard. The events are shown with their title and the exact date range. Calendar widgets that can't load or parse the calendar data will be marked as broken. How to handle broken widgets is explained in [Add and configure dashboard widgets](#) (page 59).

- **iCalendar feed source URL:** The URL of the iCalendar feed
- **Base URL:** The widget will have a link to visit a related website to see more events.
- **Number of displayed counts:** With this option, you can configure how many events should be shown in the widget.

AleksIS Logged in as admin Logout DBT

Create Icalendar Widget

Widget Title
NASA Launch Schedule

☒ Activate Widget
☐ Widget is broken

Size on mobile devices
12

<= 600 px, 12 columns
Size on tablet devices
12

> 600 px, 12 columns
Size on desktop devices
6

> 992 px, 12 columns
Size on large desktop devices
4

> 1200 px, 12 columns

ICalendar URL
http://www.nasa.gov/templateimages/redesign/calendar/iCal/nasa_calendar.ics

Base URL of related calendar
<https://www.nasa.gov/>

The widget will have a link to visit a related website to see more events

Number of displayed events
5

SAVE

Language
English

About AleksIS® – The Free School Information System © The AleksIS Team

3.9 Advanced LDAP integration

3.9.1 Interfacing with an LDAP directory beyond authentication

In addition to authenticating against an LDAP directory (as laid out in [Authenticating against LDAP](#) (page 52)), AleksIS can import personal information from LDAP. This functionality is currently limited to the information about persons and groups (cf. [The concept of persons](#) (page 39) and [The concept of groups](#) (page 40)), and related information.

Data can only be synchronised one-way. That means that, if you wish to continue maintaining personal information in LDAP, you should ensure that all changes are made in LDAP first, and then imported to AleksIS.

3.9.2 Configuring LDAP synchronisation

Setting up the LDAP synchronisation consists of three parts, which together make up the process of updating persons and groups from LDAP information.

All preferences are set under *Admin* → *Configuration* → *LDAP*.

The synchronisation always starts from a user account. Therefore, LDAP authentication needs to be set up first.

Matching fields

The first step is to configure *matching fields*. This configuration defines how persons are found in AlekSIS, based on fields in LDAP. The relevant settings are:

- **LDAP sync matching mode:** Setting this preference to *OR* means that at least one of the fields must match, whereas *AND* means all fields must match
- **LDAP sync fields:** This list defines which fields of the person in AlekSIS are considered for matching persons to LDAP entries

Field matching and rewriting

For all fields (both matching fields and other imported fields), the behaviour of the fields needs to be considered.

For every available field of a Person, the following preferences are available:

- **LDAP field for...:** This defines which LDAP attribute the data for this field is pulled from
- **Regular expression to match LDAP value for...:** If set, defines a regular expression that is applied to the attribute data from LDAP. The regular expression can contain named groups (see the [Python Documentation on Named Groups](https://docs.python.org/3/howto/regex.html#non-capturing-and-named-groups)¹⁶)
- **Replacement template to apply to...:** This template is applied to the LDAP data, and it can reference the groups matched in the regular expression defined for this field using `\g<name>`

Only fields that are configured here are honoured by the import, all other fields are ignored.

For synchronising groups, the same preferences are provided for the names and short names of the group.

Setting up what and when to synchronise

Finally, the LDAP import can be enabled by setting up the last preferences, which define when the LDAP import is run.

- **Create missing persons for LDAP users:** Defines whether persons which are not found by the matching fields are created
- **Sync LDAP user with person on login:** If this preference is enabled, persons are updated from LDAP on every login
- **Enable LDAP group sync:** If enabled, all groups the synchronised users are members of are also imported

Warning

You should take special care to thoroughly test your LDAP configuration. If operating on production data with a faulty sync configuration, important data might be overridden and destroyed.

¹⁶ <https://docs.python.org/3/howto/regex.html#non-capturing-and-named-groups>

3.9.3 Managing user accounts in LDAP

While not allowing to synchronise full personal information back into LDAP, AleksIS has limited support for managing user accounts (i.e. limited to pure authentication information) in LDAP.

Changing passwords in LDAP

When users change their password in AleksIS, it can also be changed in LDAP. This requires one of two prerequisites:

- Users must be allowed to change their own passwords in LDAP, by setting appropriate ACLs
- In the AleksIS preferences, the credentials of an account with sufficient ACLs to change all users' passwords must be configured in the respective preferences

Warning

Providing admin credentials to AleksIS imposes obvious security risks. Thus, make sure to limit this account to changing passwords. Also, make sure that other, security-critical systems which authenticate against LDAP, and AleksIS itself, require a second factor for administration, so attackers who manage to hijack an administrator account by changing its password cannot use it for anything else.

If you want to enable automatic password resets, an administrator account has to be provided in all cases, because the user triggering the password reset is not the user themselves. For more information on password resets, see [core-password-resets](#).

Creating LDAP users upon registration

If user invitations or registration are enabled, AleksIS can create the respective account in LDAP. In addition to providing admin credentials, the RDN fields that shall make up the user DN need to be configured in preferences.

For details, see [Registration](#) (page 53).

3.10 Setting up Matrix synchronization

3.10.1 Matrix and Element

[Matrix](#)¹⁷ is “an open network for secure, decentralized communication”. Being an open protocol, it allows independent implementations of servers and clients to communicate with each other. Users create their account on a server they choose and are then able to chat with everyone in the network, just like the e-mail system works.

The popular reference client [Element](#)¹⁸ with its team communication friendly user interface is a good choice for schools. It is available as a mobile app for Android and iOS, and as a web client, that can be used in a web browser.

¹⁷ <https://www.matrix.org/>

¹⁸ <https://element.io/>

3.10.2 Concepts of rooms and spaces

In Matrix, every chat is organized as a room, where people can join to send and receive messages. Schools can map their real structure to Matrix rooms to provide each group or course a place to communicate.

A Space is an organizational layer, that contains multiple rooms and can even contain other spaces. This is useful for representing child and parent group relationships.

When the Spaces feature is enabled, AleksIS will create a space for e.g. class 5a that contains the rooms 5a English, 5a Maths etc..

3.10.3 Configuring Matrix connection

To connect AleksIS to Matrix, an Account on a Matrix homeserver is required. Most homeservers of schools will not have open registration, so the process of creating an account depends on the school's setup.

Extracting the access token from Element Web

Log in to the Matrix account intended to use for AleksIS synchronization and open the settings via Profile picture → All Settings. Go to the tab Help & About and expand Access Token in the Advanced section. The access token will be displayed and can be copied.

After that, do not log out of Element, because that would invalidate the access token. You might want to use a private tab in your browser or clear session data afterwards.

Creating a new user via open registration

To create rooms and spaces and manage their participants, AleksIS needs to have a Matrix user that can be created using curl:

```
curl -XPOST -d '{"username":"aleksis", "password":"randomlygenerated",  
→ "auth": {"type":"m.login.dummy"}}' "https://example.org/_matrix/client/  
→ r0/register"
```

When the registration succeeds, a Matrix ID (user_id) and an access token will be returned:

```
{  
  "access_token": "QGV4YW1wbGU6bG9jYWxob3N0.AqdSzFmFYrLrTmteXc",  
  "home_server": "example.org",  
  "user_id": "@aleksis:example.org"  
}
```

Connecting AleksIS to the Matrix user

In AleksIS, go to Admin → Configuration → Matrix. Fill in the URL of the Matrix homeserver and the previously extracted access token.

Setting up auto-generated Matrix IDs for AleksIS users

For dedicated Matrix homeservers that use AleksIS as a SSO provider or use the same SSO provider or user database, e.g. LDAP, AleksIS can automatically generate Matrix IDs for its users based on the username.

To enable this feature, provide a value for Name of Matrix homeserver user for auto-generating Matrix IDs.

Configuring synchronization

AleksIS' Matrix synchronization creates Matrix rooms for every group and invites the respective users. Details of the synchronization can be customized with the remaining config options:

- **Disambiguate room aliases:** Room aliases are built from the group's short name or name. To avoid name collisions, a suffix will be added when this option is enabled.
- **Use Spaces:** Matrix supports grouping of rooms in so-called spaces. In a school context that can be used to create a space for each class containing rooms for all their subjects, i.e. child groups in AleksIS.
- **Power level for owners:** Specifies the Matrix room power level for group owners.
- **Power level for members:** Specifies the Matrix room power level for group members.
- **Reduce existing power levels:** For existing Matrix rooms that are going to be managed by AleksIS, power levels can be overwritten by AleksIS to match the previously defined power levels.

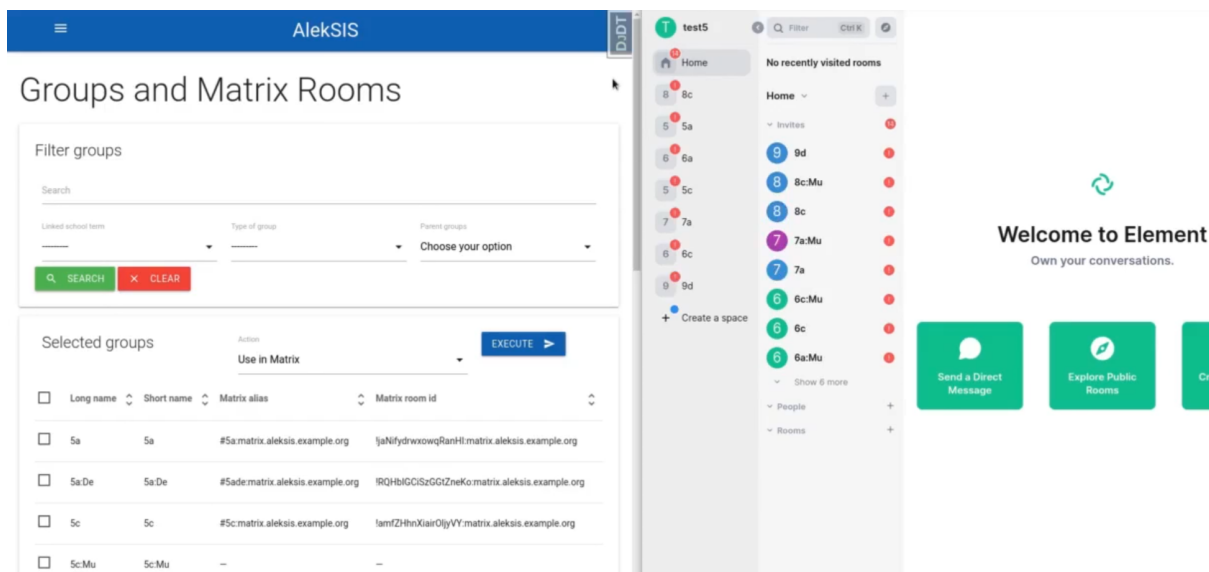
Warning

Do not change the **Use Spaces** preference after provisioning any rooms for groups with sub-groups. If you need to change the behaviour after the initial provisioning, first unlink the rooms and sketch out a manual migration plan!

3.10.4 Synchronizing with Matrix

To initially start the synchronization of groups in AleksIS to rooms, and optionally spaces, to Matrix, go to Matrix → Groups and Rooms.

Select the groups to be synchronized and click Execute. This process can take some time. All group members that have a Matrix profile in AleksIS, that may be automatically created when a Name of Matrix homeserver user for auto-generating Matrix IDs is set, get invited to the respective rooms.



The screenshot displays the AleksIS web interface. The main heading is 'Groups and Matrix Rooms'. Below it, there's a 'Filter groups' section with a search bar and dropdown menus for 'Linked school term', 'Type of group', and 'Parent groups'. A table titled 'Selected groups' shows four rows of data with columns: 'Long name', 'Short name', 'Matrix alias', and 'Matrix room id'. An 'EXECUTE' button is located to the right of the table. On the right side of the interface, there's a sidebar with a 'Home' button, a list of rooms (e.g., 8c, 5a, 6a, 5c, 7a, 6c, 9d), and a 'Welcome to Element' message with buttons for 'Send a Direct Message', 'Explore Public Rooms', and 'Create a Room'.

After this initial synchronization, every change in a Group, Matrix room or Matrix profile will be reflected in Matrix.

3.11 Support end users and get feedback with Hjelp

3.11.1 Managing FAQs

Each operator can and must write their FAQs on their own. This is done on the page *Support* → *Manage FAQ*. First of all, FAQ sections need to be created. The form for this is accessible at the top of the page. For the section to be displayed, the *Show* checkbox must be selected.

The screenshot shows the AlekSIS web interface. On the left is a sidebar with a search bar and navigation icons for Dashboard, Notifications, Account, Admin, People, and Support. The main content area is titled 'Create FAQ section'. It contains a form with the following fields: 'Name' (AlekSIS in general), 'Icon' (question_answer), and a 'Show' checkbox which is checked. Below the 'Show' checkbox is a green 'SAVE' button. At the bottom of the page, there is a blue footer bar with a 'Language' dropdown set to 'English' and the text 'About AlekSIS® — The Free School Information System © The AlekSIS Team'.

Afterwards, questions (including answers) can be created. The corresponding section must be selected. Here, too, the *Show* checkbox is important. The text in the answer field can be formatted using HTML. All icons that can be selected are located on [Material Icons](https://material.io/resources/icons/)¹⁹.

¹⁹ <https://material.io/resources/icons/>

On the overview page, the order of the questions and answers can be changed. In addition, the sections can be edited here. Hidden sections and questions are displayed darker.

3.11.2 Managing issue report form

The Hjelp issue reporting system lets the operator categorize issues in a three-level model, whereby the used categories become more concrete from the first to the third level. Every category except the first-level ones that is created therefore has to refer to a so-called parent, of which it is a subcategory.

Creating and managing categories is done in the backend admin interface under *AleksIS – Hjelp (Support) → Issue categories*.

Django administration

WELCOME, JONATHAN / VIEW SITE / CHANGE PASSWORD / LOG OUT

Home / AlekSIS – Hjelp (Support) / Issue categories

ACCOUNTS

Email addresses + Add

ALEKSIS – HJELP (SUPPORT)

FAQ questions + Add

FAQ sections + Add

Issue categories + Add

ALEKSIS – THE FREE SCHOOL INFORMATION SYSTEM

Activities + Add

Announcements + Add

Custom menu items + Add

Data check results + Add

Groups + Add

Notifications + Add

O auth access tokens + Add

O auth applications + Add

O auth grants + Add

O auth id tokens + Add



O auth refresh tokens + Add

Parental invitations + Add

Select Issue category to change

ADD ISSUE CATEGORY +

Action: 0 of 2 selected

<input type="checkbox"/>	NAME	ICON	PARENT CATEGORY	PLACEHOLDER	FREE TEXT INPUT ALLOWED
<input type="checkbox"/>	Computers		Network	Computers, notebooks	<input checked="" type="checkbox"/>
<input type="checkbox"/>	Network		-	Network-related	<input checked="" type="checkbox"/>

2 Issue categories

Upon using the *Add issue category* button, a form is displayed which contains all alterable attributes of the to-be category.

The *Name* textbox contains the text displayed when the given category is shown; and by means of the *Parent category* dropdown select list, a parent category can be selected. In case no parent category is specified, the created category is on first level.

If the *Free text input allowed* checkbox is selected, all possible children of the newly created category are ignored and instead, a free text input is displayed upon selection on the next level. One possible use case may be that the location of the selected issue has to be specified.

Special attention has to be paid to the *Icon* and *Placeholder* options as they refer to the category select dropdown/free text input of the next level. All icons that can be selected are again located on [Material Icons](https://material.io/resources/icons/)²⁰.

Django administration

WELCOME, JONATHAN / VIEW SITE / CHANGE PASSWORD / LOG OUT

Home / AlekSIS – Hjelp (Support) / Issue categories / Add Issue category

ACCOUNTS

Email addresses + Add

ALEKSIS – HJELP (SUPPORT)

FAQ questions + Add

FAQ sections + Add

Issue categories + Add

ALEKSIS – THE FREE SCHOOL INFORMATION SYSTEM

Activities + Add

Announcements + Add

Custom menu items + Add

Data check results + Add

Groups + Add

Notifications + Add

O auth access tokens + Add

O auth applications + Add

O auth grants + Add

O auth id tokens + Add



O auth refresh tokens + Add

Parental invitations + Add

Add Issue category

Name: Network

Icon: network_wifi

Parent category:  

☐ Free text input allowed

Placeholder: Network-related

Save and add another Save and continue editing SAVE

²⁰ <https://material.io/resources/icons/>

3.11.3 Settings and preferences

Hjelp requires mail delivery to be properly configured, as all three forms send their results via mail. To set the individual mail addresses the results are then sent to, one preference for each form exists:

- **Recipient e-mail address for FAQ questions**
- **Recipient e-mail address for issue reports**
- **Recipient e-mail address for feedback**

Additionally, using another preference, it can be configured whether it is possible for anonymous users to view the FAQ overview page:

- **Public visibility of FAQ**

DEVELOPER HANDBOOK

4.1 Development of AlekSIS apps using the AlekSIS core

4.1.1 Setting up the development environment

AlekSIS and all official apps use [Poetry](https://poetry.eustace.io/)²¹ to manage virtualenvs and dependencies. You should make yourself a bit comfortable with poetry by reading its documentation.

Poetry makes a lot of stuff very easy, especially managing a virtual environment that contains AlekSIS and everything you need to run the framework and selected apps. The minimum supported version of Poetry is 1.2.0.

Also, [Yarn](https://yarnpkg.com)²² is needed to resolve JavaScript dependencies.

For repository management, *myrepos* is required.

Setup database and message broker

AlekSIS requires [PostgreSQL](https://www.postgresql.org/)²³ (version 15 or newer) as database backend. It requires the `pg-rrule` extension, which can be installed from the PostgreSQL APT repository. To provide a database named *aleksis* with a user named *aleksis* on Debian:

```
sudo apt install postgresql-common
sudo /usr/share/postgresql-common/pgdg/apt.postgresql.org.sh -y
sudo apt install postgresql-17 postgresql-17-pg-rrule
sudo -u postgres createuser -P aleksis
sudo -u postgres createdb -O aleksis aleksis
sudo -u postgres psql -c "CREATE EXTENSION pg_rrule" aleksis
```

Additionally, [Valkey](https://valkey.io/)²⁴ is used as message broker and for caching. On some distributions, the legacy Redis broker is still used instead. The default configuration of the server in Debian is sufficient:

```
sudo apt install redis-server
```

²¹ <https://poetry.eustace.io/>

²² <https://yarnpkg.com>

²³ <https://www.postgresql.org/>

²⁴ <https://valkey.io/>

Get the source tree

To download AleksIS and all officially bundled apps in their development version, use Git like so:

```
git clone https://edugit.org/AleksIS/official/AleksIS
```

This first downloads a meta repository that contains a config file for mr. To clone the AleksIS-Core and all official (and onboarding) apps, run:

```
mr update
```

Install native dependencies

Some system libraries are required to install AleksIS. On Debian, for example, this would be done with:

```
sudo apt install build-essential libpq-dev libpq5 libssl-dev python3-dev  
↳python3-pip python3-venv yarnpkg gettext firefox-esr
```

Get Poetry

Make sure to have Poetry installed like described in its documentation. Right now, we encourage using pip to install Poetry once system-wide (this will change once distributions pick up Poetry):

```
sudo pip3 install poetry
```

You can use any other of the [Poetry installation methods](#)²⁵.

Install AleksIS in its own virtual environment

Poetry will automatically manage virtual environments per project, so installing AleksIS is a matter of switching into the Core's directory and running the initial AleksIS installation:

```
cd apps/official/AleksIS-Core  
poetry install
```

Now it's recommended to run a shell that uses the newly created venv:

```
poetry shell
```

Regular tasks

After making changes to the environment, e.g. installing apps or updates, some maintenance tasks need to be done:

1. Download and install JavaScript dependencies
2. Compile SCSS files
3. Collect static files
4. Compile translation strings
5. Run database migrations

²⁵ <https://poetry.eustace.io/docs/#installation>

6. Create initial revisions (for django-reversion)

All six steps can be done with the `poetry shell` command and `aleksis-admin`:

```
ALEKSIS_maintenance__debug=true ALEKSIS_database__password=aleksis poetry_
→shell
poetry run aleksis-admin vite build
poetry run aleksis-admin compile_scss
poetry run aleksis-admin collectstatic --clear
poetry run aleksis-admin compilemessages
poetry run aleksis-admin migrate
poetry run aleksis-admin createinitialrevisions
```

Running the development server

The development server can be started using Django's `runserver` command. If you want to automatically start other necessary tools in development, like the [Celery](#)²⁶ worker and scheduler, use `runuwsgi` instead. You can either configure AleksIS like in a production environment, or pass basic settings in as environment variable. Here is an example that runs the development server against a local PostgreSQL database with password *aleksis* (all else remains default) and with the *debug* setting enabled:

```
ALEKSIS_maintenance__debug=true ALEKSIS_database__password=aleksis poetry_
→run aleksis-admin runuwsgi
```

4.1.2 Installing apps into development environment

Officially bundled apps

Officially bundled apps are available in the `apps/official/` sub-folder of the meta repository. If you followed the documentation, they will already be checked out in the version required for the bundle you are running.

Installing a development environment for own apps

If you are developing your own app, you probably do not want to run a development environment from the *AleksIS-Core* repository.

Instead, simply install the environment using `poetry install` from your app repository – it will pull in *AleksIS-Core* as a dependency automatically, and everything will work as described beforehand.

Note

Take care not to mix up environments, especially if using `poetry shell`.

²⁶ <https://celeryproject.org/>

Using one virtual environment for everything

Note

This method is not encouraged for most use cases.

Installing apps into the existing virtual environment of *AlekSIS-Core* can be easily done after starting *poetry shell*:

```
poetry install
```

Do not forget to run the maintenance tasks described earlier after installing any app.

Note

This is not suitable for working on the core, because it will install the *AlekSIS-Core* version used by the app using *pip* again.

4.1.3 Running tests and reports

Running default test suite

The test suite can be run using the *tox* tool:

```
tox
```

Enabling Selenium browser tests

The test suite contains tests that use Selenium to do browser based tests. They need to be enabled when running the test suite, which can be done by setting certain environment variables:

Variable	Meaning	Example
TEST_SELENIUM_BROWSERS	List of webdrivers to test against, comma-separated.	chrome,firefox
TEST_SELENIUM_HUB	Address of Selenium hub if using remote grid	http://127.0.0.1:4444/wd/hub
TEST_HOST	Hostname reachable from Selenium for live server	172.17.0.1
TEST_SCREENSHOT_PATH	Path to directory to create screenshots in	./screenshots

Selenium tests are enabled if *TEST_SELENIUM_BROWSERS* is non-empty.

To set variables, use *env* to wrap the *tox* command:

```
TEST_SELENIUM_BROWSERS=chrome,firefox tox
```

Using a Selenium hub on local Docker host

One way to setup Selenium is to use the official images on the local machine.

First, get Selenium Hub and one or more browser nodes up and running:

```
docker run -d -p 4444:4444 --name selenium-hub selenium/hub
docker run -d --link selenium-hub:hub selenium/node-chrome
docker run -d --link selenium-hub:hub selenium/node-firefox
```

After that, you can run the test suite, setting the needed variables to use Docker Hub:

```
TEST_SELENIUM_BROWSERS=chrome,firefox \
TEST_SELENIUM_HUB=http://127.0.0.1:4444/wd/hub \
TEST_HOST=172.17.0.1 \
tox
```

The `TEST_HOST` variable is set to the Docker host's IP address, where the Selenium nodes can access Django's live server. Django automatically configures the live server to be reachable if a Selenium hub is used.

Taking screenshots

The browser test suites automatically take screenshots at certain steps if enabled in the test run. This can be used to visually verify that views look like they should or for documentation purposes.

To enable screenshots, add the `TEST_SCREENSHOT_PATH` environment variable when running the tests.

If running multiple browsers, screenshots are placed in separate directories per browser.

4.1.4 Materialize templates

The legacy AleksIS frontend uses the [MaterializeCSS](https://materializecss.com/)²⁷ framework and the [django-material](https://pypi.org/project/django-material/)²⁸ library.

Version 3.0 introduced a dynamic frontend app written in `'Vue.js'` which communicates with the backend through `'GraphQL'`. Support for legacy views (Django templates and Materialize) was removed; while there is backwards compatibility for now, this is only used by official apps until their views are fully migrated. Then, backwards compatibility will be entirely removed.

Internationalisation

Load the `i18n` template tag and start translating strings in templates with the following template tags:

```
{% blocktrans %}String{% endblocktrans %}
{% trans "String" %}
```

`{% blocktrans %}` is mostly used for multiple words or multiline, while `{% trans %}` is used for single words.

²⁷ <https://materializecss.com/>

²⁸ <https://pypi.org/project/django-material/>

Title and headlines

To add a main headline or browser title to your template, you can add the following blocks to your template:

```
{% block browser_title %}Title{% endblock %}
{% block page_title %}Headline{% endblock %}
```

To fully remove page or browser title, use these template tags:

```
{% block no_browser_title %}{% endblock %}
{% block no_page_title %}{% endblock %}
```

Extended navbar

The top navbar with the site title and the login status can be extended by an additional tab bar, for example.

To add normal Materialize tabs without icons, you can use a snippet like described in [Extended Navbar with Tabs](#)²⁹:

```
{% block nav_content %}
  <ul class="tabs tabs-transparent">
    <li class="tab"><a href="#test1">Test 1</a></li>
    <li class="tab"><a class="active" href="#test2">Test 2</a></li>
    <li class="tab disabled"><a href="#test3">Disabled Tab</a></li>
    <li class="tab"><a href="#test4">Test 4</a></li>
  </ul>
{% endblock %}
```

Furthermore, you can use tabs with integrated icons that are higher, but more compact in width:

```
{% block nav_content %}
  <ul class="tabs tabs-transparent tabs-icons tabs-fixed-width">
    <li class="tab">
      <a href="#test1">
        <i class="material-icons iconify" data-icon="mdi:speaker_
→notes"></i>
        Test 1
      </a>
    </li>
    <li class="tab">
      <a href="#test2">
        <i class="material-icons iconify" data-icon="mdi:people"></
→i>
        Test 2
      </a>
    </li>
  </ul>
{% endblock %}
```

²⁹ <https://materializecss.com/navbar.html#navbar-tabs>

⚠ Warning

Don't define the `nav_content` block within the content block – that won't work.

Forms in templates

The django MaterializeCSS integrations provides support for forms in template.

You just have to load the `material_form` templatetag in the `{% load %}` block.

The following snippet generates the form:

```
<form method="post" enctype="multipart/form-data">
  {% csrf_token %}
  {% form form=edit_person_form %}{% endform %}
  {% include "core/partials/save_button.html" %}
</form>
```

`edit_person_form` is the variable name of the form in your context.

`{% include "core/partials/save_button.html" %}` includes a template snippet from AleksIS core. You can modify the buttons icon and translatable caption like this:

```
{% trans "Edit" as caption %}
{% include "core/partials/save_button.html" with caption=caption, icon=
  → "person" %}
```

In your `forms.py` you can configure the layout of the fields like in the `EditPersonForm`:

```
class EditPersonForm(ExtensibleForm):
    """Form to edit an existing person object in the frontend."""

    layout = Layout(
        Fieldset(
            _("Base data"),
            "short_name",
            Row("user", "primary_group"),
            Row("first_name", "additional_name", "last_name"),
        ),
        Fieldset(_("Address"), Row("street", "houenumber"), Row("postal_code",
        → "place")),
        Fieldset(_("Contact data"), "email", Row("phone_number", "mobile_number
        →")),
        Fieldset(
            _("Advanced personal data"), Row("sex", "date_of_birth"), Row(
        → "photo"), "guardians",
        ),
    )
```

Tables in templates

To display tables generated by `django-tables2` in your template, you have to load the `render_table` template tag from `django_tables2`:

```
{% load render_table from django_tables2 %}
```

After you've loaded the template tag, you can simply generate the table like this:

```
{% render_table persons_table %}
```

`persons_table` is the variable name of the table in your context.

4.1.5 Vue-based frontend

The AlekSIS frontend is based on the `'Vue.js'` framework and the `'Vuetify'` UI library. It communicates with the backend using a *GraphQL* API.

4.1.6 Extensible models

In AlekSIS you can use `ExtensibleModels` to add model fields to other apps models.

If you want to make your apps models extensible, use the `ExtensibleModel` class as parent class of your models.

4.1.7 Merging of app settings

AlekSIS provides features to merge app settings into main `settings.py`.

Third-party apps can only add values to some select existing settings. Official apps (those under the `aleksis.apps.namespace`) can mark any setting for overriding.

Currently mergable settings

The following settings can be amended by any app:

- `INSTALLED_APPS`
- `DATABASES`
- `YARN_INSTALLED_APPS`
- `ANY_JS`

If you want to add another database for your AlekSIS app, you have to add the following into your `settings.py`:

```
DATABASES = {
    "database": {
        "ENGINE": "django.db.backends.postgresql",
        "NAME": "database",
        "USER": "database",
        "PASSWORD": "Y0urV3ryR4nd0mP4ssw0rd",
        "HOST": "127.0.0.1",
        "PORT": 5432,
    }
}
```

Overriding any setting

Official apps only (currently) can override any setting, but need to explicitly mark it by listing it in a list called `overrides` in their `settings.py`:

```
PAYMENT_MODEL = "tezor.Invoice"

overrides = ["PAYMENT_MODEL"]
```

4.1.8 Registering dashboard widgets

Apps can register their own dashboard widgets which are automatically registered in the corresponding frontend for configuring them.

To implement a widget, add a model that subclasses `DashboardWidget`, set the template and implement the `get_context` method to return a dictionary to be passed as context to the template. The template system works as in every Django view and allows you to use the normal Django template language.

If your widget does not add any custom database fields, you should mark it as a proxy model.

You can provide a `Media` meta class with custom JS and CSS files which will be added to the HTML head on the dashboard if the dashboard widget is shown. For further information on media definition, see [Django Media](#)³⁰.

Example:

```
from django.forms.widgets import Media

from aleksis.core.models import DashboardWidget

class MyWidget(DashboardWidget):
    template = "myapp/widget.html"

    def get_context(self, request):
        context = {"some_content": "foo"}
        return context

    class Meta:
        proxy = True

    media = Media(css={
        'all': ('pretty.css',)
    },
    js=('animations.js', 'actions.js')
    )
```

³⁰ <https://docs.djangoproject.com/en/3.0/topics/forms/media/>

4.2 Providing auto-generated documents from apps

4.2.1 Providing live document types from other apps

AleksIS apps can provide live document types that are then managed by Resint. Live document types are Django models sub-classing the LiveDocument model, and providing fields and some methods that define how the live document is generated.

The following stripped-down example shows how to provide a live document type.

```
from aleksis.apps.resint.models import LiveDocument

class AutomaticPlan(LiveDocument):
    # Template name to render
    template = "my_app/pdf_template.html"

    # A field to be rendered in the edit form
    number_of_days = models.PositiveIntegerField(
        default=1,
    )

    def get_context_data(self) -> Dict[str, Any]:
        """Get context data to pass to the PDF template."""
        context = {}
        # Do something ehre to construct the context data
        return context
```

This basic example generates a PDF by defining an HTML template and overriding the `get_context_data` method. This method has to return a context dictionary, which will then be passed to the template.

If you need more control over how the PDF is generated, you can instead override the `update` method:

```
class AutomaticPlan(LiveDocument):
    def update(self, triggered_manually: bool = True):
        """Re-generate PDF for this live document."""

        # Do whatever is necessary to get file contents

        self.current_file.save(self.filename, content)
```

You need to ensure that `update()` is called whenever you need to provide a new version of your document. One possibility is to listen to some relevant Django signal, then call `update()` if necessary.

5.1 About Free Software and Open Source

AlekSIS®, as explained in the [Introduction to AlekSIS](#) (page 1), is a School Information System that is *free software*. This means that the software can be freely used by anyone who wants it, and a few other things.

The terms *free software* and *open source* are regularly used interchangeably. The term *free software* emphasises the terminology of the granted freedoms better.

5.1.1 The four freedoms

Free software is defined by the *four freedoms*, which describe what rights users of free software are entitled to. The word *free* thus is meant as in *free speech*, not as in *free drink* – it means that the software is accompanied by certain freedoms [[FreeSoftware](#)].

Freedom to use

Free software can be used for any purpose. There are no limitations imposed by the authors. For example, the software may be used both in school and at home, without having to acquire a separate licence.

Freedom to study

The source code of free software can be read. Free software is always shipped with the source code it was written in. This means that users can verify how the software works, what it does, and learn from the code.

Freedom to share

The licence for free software can be passed on. Everyone who has a copy of free software is allowed to share it with everyone they want, and the others receive the same rights and freedoms. This means that a school who acquired some software is allowed to pass it on to teachers, students, and their families, without having to acquire new licences.

Freedom to improve

The code of free software can be modified in order to improve it, more make new software based on existing. The result can be shared with others (sometimes, the software needs to be given a new name to reduce confusion). Most free software projects also accept improvements from users, and incorporate them into their work.

5.1.2 Advantages for schools

Educational institutions draw several benefits from the freedoms described above. In essence, using free software is a natural extension of independent, pluralistic education.

Sustainability

Because free software authors do not sell licences, they do not benefit from selling new copies of their software. Furthermore, they do not have contracts with hardware vendors. Commercial vendors often make changes to their software that make their customers buy newer, faster copies of the software, or buy new hardware from a partner.

Free software often works on older hardware, prolonging the life of devices and reducing costs.

Sharing, copying and reusability

Because free software does not restrict in what scenarios it may be used or how many copies may be made, the same software can easily be used by students and teachers at home. There is no need to buy more licenses from the school budget, or to make students buy licences from their private budget.

Reduced vendor lock-in

Commercial software vendors benefit from selling as many copies of their software as possible, and from doing so for a long time. Therefore, they try to turn their products into a self-defined industry standard by getting students accustomed to their software as warly as possible. Because they often do not provide open interfaces or good compatibility with other products, escaping from this vendor lock-in is difficult.

5.2 Legal Information about the AlekSIS® Software

5.2.1 Copyright Note and Licenses

AlekSIS® Core and official apps

The AlekSIS Core and all official apps are copyright by the respective developers, as listed in the text files accompanying each source tree. They are licensed under the terms of the European Union Public License (EUPL), version 1.2 or later.

The AlekSIS® Handbook

This handbook itself, with the exception of the cover artwork, is copyright 2021 by the authors listed on the cover, and licensed under the EUPL, version 1.2 or later.

The cover artwork remix, as used in the LaTeX/PDF handbook, is copyright 2021 by Dominik George, and was set in LaTeX by mirabilos in 2022.

The following cover artwork photos are copyright 2021 by Max Knopp of Katharineum zu Lübeck:

- Children in front of digital signage monitor
- Child's hand with smartphone
- Teachers in front of computer screen

The following cover artwork photo is copyright 2020 Martin Gummi of Teckids e.V.:

- People in computer lab with laptops

All artwork is licensed under EUPL 1.2 or later or, optionally, under CC-BY-SA 4.0 or later.

5.3 AleksIS® trademark

AleksIS® is a registered trademark of the AleksIS open source project, represented by [Teckids e.V.](#)³¹; use of the trademark is only permitted in accordance with the trademark policy published herein.

Without express permission and without additional remarks, the trademark AleksIS® can be used for the following purposes:

- Description of the original software, as published in the code repositories and on the websites of the AleksIS® open source project
- Description of the open source project, the team and its goals, especially in journalistic publications and presentations

With explicit remarks, the trademark AleksIS® can be used for the following purposes:

- Description of derivative works
- Description of services and commercial offers

Users of the trademark have to choose wording that distinctly highlights the relation to the AleksIS® open source project (e.g. “based on AleksIS®”) and are required to add the remark “AleksIS® is a registered trademark of the AleksIS open source project, represented by Teckids e.V.”.

Changes of solely technical nature, e.g. patches added by distribution vendors to adjust AleksIS to their infrastructure, for integration with other software packages, or to add unpublished bugfixes that do not alter the functionality of the software in essential ways discernable by end users, do not constitute a derivative work for the purpose of this trademark policy.

[Teckids e.V.](#)³² reserves the right to license use of the trademark on request.

5.4 EUROPEAN UNION PUBLIC LICENCE v. 1.2

EUPL © the European Union 2007, 2016

This European Union Public Licence (the ‘EUPL’) applies to the Work (as defined below) which is provided under the terms of this Licence. Any use of the Work, other than as authorised under this Licence is prohibited (to the extent such use is covered by a right of the copyright holder of the Work).

The Work is provided under the terms of this Licence when the Licensor (as defined below) has placed the following notice immediately following the copyright notice for the Work:

Licensed under the EUPL

or has expressed by any other means his willingness to license under the EUPL.

5.4.1 1. Definitions

In this Licence, the following terms have the following meaning:

- ‘The Licence’: this Licence.

³¹ <https://teckids.org/>

³² <https://teckids.org/>

- 'The Original Work': the work or software distributed or communicated by the Licensor under this Licence, available as Source Code and also as Executable Code as the case may be.
- 'Derivative Works': the works or software that could be created by the Licensee, based upon the Original Work or modifications thereof. This Licence does not define the extent of modification or dependence on the Original Work required in order to classify a work as a Derivative Work; this extent is determined by copyright law applicable in the country mentioned in Article 15.
- 'The Work': the Original Work or its Derivative Works.
- 'The Source Code': the human-readable form of the Work which is the most convenient for people to study and modify.
- 'The Executable Code': any code which has generally been compiled and which is meant to be interpreted by a computer as a program.
- 'The Licensor': the natural or legal person that distributes or communicates the Work under the Licence.
- 'Contributor(s)': any natural or legal person who modifies the Work under the Licence, or otherwise contributes to the creation of a Derivative Work.
- 'The Licensee' or 'You': any natural or legal person who makes any usage of the Work under the terms of the Licence.
- 'Distribution' or 'Communication': any act of selling, giving, lending, renting, distributing, communicating, transmitting, or otherwise making available, online or offline, copies of the Work or providing access to its essential functionalities at the disposal of any other natural or legal person.

5.4.2 2. Scope of the rights granted by the Licence

The Licensor hereby grants You a worldwide, royalty-free, non-exclusive, sublicensable licence to do the following, for the duration of copyright vested in the Original Work:

- use the Work in any circumstance and for all usage,
- reproduce the Work,
- modify the Work, and make Derivative Works based upon the Work,
- communicate to the public, including the right to make available or display the Work or copies thereof to the public and perform publicly, as the case may be, the Work,
- distribute the Work or copies thereof,
- lend and rent the Work or copies thereof,
- sublicense rights in the Work or copies thereof.

Those rights can be exercised on any media, supports and formats, whether now known or later invented, as far as the applicable law permits so.

In the countries where moral rights apply, the Licensor waives his right to exercise his moral right to the extent allowed by law in order to make effective the licence of the economic rights here above listed.

The Licensor grants to the Licensee royalty-free, non-exclusive usage rights to any patents held by the Licensor, to the extent necessary to make use of the rights granted on the Work under this Licence.

5.4.3 3. Communication of the Source Code

The Licensor may provide the Work either in its Source Code form, or as Executable Code. If the Work is provided as Executable Code, the Licensor provides in addition a machine-readable copy of the Source Code of the Work along with each copy of the Work that the Licensor distributes or indicates, in a notice following the copyright notice attached to the Work, a repository where the Source Code is easily and freely accessible for as long as the Licensor continues to distribute or communicate the Work.

5.4.4 4. Limitations on copyright

Nothing in this Licence is intended to deprive the Licensee of the benefits from any exception or limitation to the exclusive rights of the rights owners in the Work, of the exhaustion of those rights or of other applicable limitations thereto.

5.4.5 5. Obligations of the Licensee

The grant of the rights mentioned above is subject to some restrictions and obligations imposed on the Licensee. Those obligations are the following:

Attribution right: The Licensee shall keep intact all copyright, patent or trademarks notices and all notices that refer to the Licence and to the disclaimer of warranties. The Licensee must include a copy of such notices and a copy of the Licence with every copy of the Work he/she distributes or communicates. The Licensee must cause any Derivative Work to carry prominent notices stating that the Work has been modified and the date of modification.

Copyleft clause: If the Licensee distributes or communicates copies of the Original Works or Derivative Works, this Distribution or Communication will be done under the terms of this Licence or of a later version of this Licence unless the Original Work is expressly distributed only under this version of the Licence – for example by communicating ‘EUPL v. 1.2 only’. The Licensee (becoming Licensor) cannot offer or impose any additional terms or conditions on the Work or Derivative Work that alter or restrict the terms of the Licence.

Compatibility clause: If the Licensee Distributes or Communicates Derivative Works or copies thereof based upon both the Work and another work licensed under a Compatible Licence, this Distribution or Communication can be done under the terms of this Compatible Licence. For the sake of this clause, ‘Compatible Licence’ refers to the licences listed in the appendix attached to this Licence. Should the Licensee’s obligations under the Compatible Licence conflict with his/her obligations under this Licence, the obligations of the Compatible Licence shall prevail.

Provision of Source Code: When distributing or communicating copies of the Work, the Licensee will provide a machine-readable copy of the Source Code or indicate a repository where this Source will be easily and freely available for as long as the Licensee continues to distribute or communicate the Work. *Legal Protection:* This Licence does not grant permission to use the trade names, trademarks, service marks, or names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the copyright notice.

5.4.6 6. Chain of Authorship

The original Licensor warrants that the copyright in the Original Work granted hereunder is owned by him/her or licensed to him/her and that he/she has the power and authority to grant the Licence.

Each Contributor warrants that the copyright in the modifications he/she brings to the Work are owned by him/her or licensed to him/her and that he/she has the power and authority to grant the Licence.

Each time You accept the Licence, the original Licensor and subsequent Contributors grant You a licence to their contributions to the Work, under the terms of this Licence.

5.4.7 7. Disclaimer of Warranty

The Work is a work in progress, which is continuously improved by numerous Contributors. It is not a finished work and may therefore contain defects or 'bugs' inherent to this type of development. For the above reason, the Work is provided under the Licence on an 'as is' basis and without warranties of any kind concerning the Work, including without limitation merchantability, fitness for a particular purpose, absence of defects or errors, accuracy, non-infringement of intellectual property rights other than copyright as stated in Article 6 of this Licence.

This disclaimer of warranty is an essential part of the Licence and a condition for the grant of any rights to the Work.

5.4.8 8. Disclaimer of Liability

Except in the cases of wilful misconduct or damages directly caused to natural persons, the Licensor will in no event be liable for any direct or indirect, material or moral, damages of any kind, arising out of the Licence or of the use of the Work, including without limitation, damages for loss of goodwill, work stoppage, computer failure or malfunction, loss of data or any commercial damage, even if the Licensor has been advised of the possibility of such damage. However, the Licensor will be liable under statutory product liability laws as far such laws apply to the Work.

5.4.9 9. Additional agreements

While distributing the Work, You may choose to conclude an additional agreement, defining obligations or services consistent with this Licence. However, if accepting obligations, You may act only on your own behalf and on your sole responsibility, not on behalf of the original Licensor or any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against such Contributor by the fact You have accepted any warranty or additional liability.

5.4.10 10. Acceptance of the Licence

The provisions of this Licence can be accepted by clicking on an icon 'I agree' placed under the bottom of a window displaying the text of this Licence or by affirming consent in any other similar way, in accordance with the rules of applicable law. Clicking on that icon indicates your clear and irrevocable acceptance of this Licence and all of its terms and conditions.

Similarly, you irrevocably accept this Licence and all of its terms and conditions by exercising any rights granted to You by Article 2 of this Licence, such as the use of the Work, the creation by You of a Derivative Work or the Distribution or Communication by You of the Work or copies thereof.

5.4.11 11. Information to the public

In case of any Distribution or Communication of the Work by means of electronic communication by You (for example, by offering to download the Work from a remote location) the distribution channel or media (for example, a website) must at least provide to the public the information requested by the applicable law regarding the Licensor, the Licence and the way it may be accessible, concluded, stored and reproduced by the Licensee.

5.4.12 12. Termination of the Licence

The Licence and the rights granted hereunder will terminate automatically upon any breach by the Licensee of the terms of the Licence.

Such a termination will not terminate the licences of any person who has received the Work from the Licensee under the Licence, provided such persons remain in full compliance with the Licence.

5.4.13 13. Miscellaneous

Without prejudice of Article 9 above, the Licence represents the complete agreement between the Parties as to the Work.

If any provision of the Licence is invalid or unenforceable under applicable law, this will not affect the validity or enforceability of the Licence as a whole. Such provision will be construed or reformed so as necessary to make it valid and enforceable.

The European Commission may publish other linguistic versions or new versions of this Licence or updated versions of the Appendix, so far this is required and reasonable, without reducing the scope of the rights granted by the Licence.

New versions of the Licence will be published with a unique version number.

All linguistic versions of this Licence, approved by the European Commission, have identical value. Parties can take advantage of the linguistic version of their choice.

5.4.14 14. Jurisdiction

Without prejudice to specific agreement between parties,

- any litigation resulting from the interpretation of this License, arising between the European Union institutions, bodies, offices or agencies, as a Licensor, and any Licensee, will be subject to the jurisdiction of the Court of Justice of the European Union, as laid down in article 272 of the Treaty on the Functioning of the European Union,
- any litigation arising between other parties and resulting from the interpretation of this Licence, will be subject to the exclusive jurisdiction of the competent court where the Licensor resides or conducts its primary business.

5.4.15 15. Applicable Law

Without prejudice to specific agreement between parties,

- this Licence shall be governed by the law of the European Union Member State where the Licensor has his seat, resides or has his registered office,
- this licence shall be governed by Belgian law if the Licensor has no seat, residence or registered office inside a European Union Member State.

5.4.16 Appendix

'Compatible Licences' according to Article 5 EUPL are:

- GNU General Public License (GPL) v. 2, v. 3
- GNU Affero General Public License (AGPL) v. 3
- Open Software License (OSL) v. 2.1, v. 3.0
- Eclipse Public License (EPL) v. 1.0
- CeCILL v. 2.0, v. 2.1
- Mozilla Public Licence (MPL) v. 2
- GNU Lesser General Public Licence (LGPL) v. 2.1, v. 3
- Creative Commons Attribution-ShareAlike v. 3.0 Unported (CC BY-SA 3.0) for works other than software
- European Union Public Licence (EUPL) v. 1.1, v. 1.2
- Québec Free and Open-Source Licence – Reciprocity (LiLiQ-R) or Strong Reciprocity (LiLiQ-R+)

The European Commission may update this Appendix to later versions of the above licences without producing a new version of the EUPL, as long as they provide the rights granted in Article 2 of this Licence and protect the covered Source Code from exclusive appropriation.

All other changes or additions to this Appendix require the production of a new EUPL version.

BIBLIOGRAPHY

- [UnixPhil] Raymond, Eric S. (2004). "Basics of the Unix Philosophy". The Art of Unix Programming. Addison-Wesley Professional (published 2003-09-23). ISBN 0-13-142901-9.
- [ConnectionPool] connection pooling options, psycopg2 docs, https://www.psycopg.org/psycopg3/docs/api/pool.html#psycopg_pool.ConnectionPool
- [FreeSoftware] "What is Free Software", Free Software Foundation Europe, <https://fsfe.org/freesoftware/>

