Zürcher Hochschule
für Angewandte Wissenschaften

School of
Engineering

InIT Institut für angewandte
Informationstechnologie

Bachelorarbeit Informatik

ZHAWo
Platform Independent Timetable App

Autoren	Dominik Bachmann Julian Visser
Hauptbetreuung	Andreas Meier
Datum	05.06.2019

Zürcher Hochschule
für Angewandte Wissenschaften

School of
Engineering

Erklärung betreffend das selbständige Verfassen einer Bachelorarbeit an der School of Engineering

Mit der Abgabe dieser Bachelorarbeit versichert der/die Studierende, dass er/sie die Arbeit selbständig und ohne fremde Hilfe verfasst hat. (Bei Gruppenarbeiten gelten die Leistungen der übrigen Gruppenmitglieder nicht als fremde Hilfe.)

Der/die unterzeichnende Studierende erklärt, dass alle zitierten Quellen (auch Internetseiten) im Text oder Anhang korrekt nachgewiesen sind, d.h. dass die Bachelorarbeit keine Plagiate enthält, also keine Teile, die teilweise oder vollständig aus einem fremden Text oder einer fremden Arbeit unter Vorgabe der eigenen Urheberschaft bzw. ohne Quellenangabe übernommen worden sind.

Bei Verfehlungen aller Art treten die Paragraphen 39 und 40 (Unredlichkeit und Verfahren bei Unredlichkeit) der ZHAW Prüfungsordnung sowie die Bestimmungen der Disziplinarmassnahmen der Hochschulordnung in Kraft.

Ort, Datum:

Winterthur, 05.06.2019

Unterschriften:

[illegible]

[illegible]

Das Original dieses Formulars ist bei der ZHAW-Version aller abgegebenen Bachelorarbeiten zu Beginn der Dokumentation nach dem Titelblatt mit Original-Unterschriften und -Datum (keine Kopie) einzufügen.

Contents

Abstract

With ZHAWo we want to provide students of the Zurich University of Applied Sciences (ZHAW) with a single application for their daily study tasks. Students need to be able to quickly look up their timetable and what lunch menus are available at their campus mensa. While these functions are provided by the university's official app, we want to provide a consistent user experience among different platforms using Progressive Web App technologies. Additionally, ZHAWo offers an interface to search for free rooms. This function was previously provided by an app that is no longer available and was extensively used to find rooms for projects or study groups to work in environment less busy than the usually crowded official work spaces. We also integrated news and events from the Verein Studierende ZHAW (vszhaw) into ZHAWo to provide students with easier access to information about events and services. A student survey confirmed the high demand for the room search feature. Our application received good ratings for usability, performance and design.
As a secondary objective, we evaluated the practicability of using cross platform web technologies to build a Progressive Web App. It enabled us to apply an agile development approach with fast prototyping of new features while still providing an application with the functionalities and feel that users expect from a native app. With this approach, we also avoided having to maintain multiple code bases for different platforms and mixing different technologies for frontend and backend. Progressive Web App technology is still very new. There are notable differences in support and implementation. As the the technology matures, we expect it to be a viable alternative to building native applications.

1 Introduction

With ZHAWo [1], our goal is to provide students of the Zurich University of Applied Sciences (ZHAW) with an application that allows them to access their timetable and mensa menus in one single cross-platform application. Additionally, students can look for unoccupied rooms for both group projects and a quiet workspace. This functionality was - until about two years ago - provided by an Android application that was no longer maintained and eventually disappeared from the Google Play Store. While the official study rooms at the Technikum campus offer a space for both quiet work as well as group projects, in our experience as students it was very convenient to have a service to quickly look up free rooms without having to walk from room to room. Another feature ZHAWo provides is the integration of news [2] and events [3] of the vszhaw directly into the application. We aim to reduce the effort that is needed to stay up to date with the vszhaw and hope that both students and the vszhaw can profit from this.
Students have to visit multiple websites or use different applications on a daily basis in order to get information about their timetable, the offered menus in their campus mensa and events organized by the Verein Studierende ZHAW (vszhaw).
For their timetable, a student can either visit the official site [4] or use the official CampusInfo application for either Android [5] or iOS [6]. The official site was designed for use on desktop browsers and is not optimised for responsiveness and display on phones. And while the official Android application is well maintained and offers a good user experience and a lot of additional features - such as direct access to public transport timetables and mensa menu plans - its iOS counterpart was seemingly lagging behind in development and at least at the start of this project did not offer the same user experience. The biggest issue with the iOS application was the lack of offline functionality. When a user's network cut out, even the timetable information that was previously loaded could no longer be accessed after navigating away. The iOS application has since received an update with a much improved design and offline functionality. This difference in quality and features is a common occurrence because the development of native applications requires two separate code bases for Android and iOS.
When students want to check the mensa menus of their campus, they have the option of visiting the SV groups site [7] or to use the official Android or iOS application. These options suffered from the same issues as previously explained for the schedule and were also improved in a recent update.
The goal of this project was to build a production-ready application that can be distributed to and used by the students. The focus is on development of a PWA while evaluating advantages and disadvantages of using Progressive Web App technologies as well as user reception of our application.
By using Progressive Web Application (PWA) technologies [8] in combination with JavaScript frameworks for both front- and backend, we achieved a consistent user experience on desktop, Android and iOS devices. We eliminated the issue of having to maintain separate code bases for different platforms while still being able to provide a native feel and functionality. PWA features such as offline caching of HTTP requests allow us to overcome previously mentioned issues with reliability on spotty networks. An additional advantage we gained by using PWA technologies and the same programming language across the full stack, was fast prototyping in an agile development process. Development of additional features and functionality can be achieved at a much faster rate with a single code base across all platforms.

2 Development

2.1 Agile Approach

For the development of ZHAWo we used an iterative agile approach. The code base was managed on a single GitHub repository [9] for both front- and backend. User stories were tracked using GitHub issues and the sprints were organised using GitHub project boards. We structured the development into sprints of 2 weeks. We used a small group of students to regularly test our application and provide us with feedback. This helped with identifying the most important issues with our implementation of the user stories. For example, one feature that was regularly requested was to be able to use swipe gestures to navigate between different days. We adjusted our planning accordingly and improved the application based on the feedback.
The flexibility of using only JavaScript and PWA technologies enabled us to rapidly prototype new ideas without having to maintain two different code bases. User stories were implemented and tested in feature branches and merged into the master branch as part of the sprint reviews. This practice ensured that we had stable iterations of the application to deploy for user feedback. A detailed protocol of our sprint planning can be found on our GitHub wiki [10].

2.2 Continuous Integration & Deployment

For continuous integration we used Travis CI [11] in combination with Codecov [12] to track test coverage. With good integration of both these tools into GitHub - for example reporting of build status and test coverage changes in GitHub comments on each pull request - we ensured that no breaking changes were introduced into the production build and were able to monitor our test coverage. Each new iteration was deployed to a ZHAW server [1] and was directly pushed to our users on their next visit, without having to install an update through an app store as it would be the case with native applications.

2.3 Primary Functions

To structure the development and implementation, we established four primary functions that our application should provide.

- **Timetable**: A student can access their timetable and look up timetables of lecturers, classes, courses and rooms.
- **Menu plans**: A student can access menu plans of the different campus mensas across the ZHAW.
- **Room search**: A student can look for and find free rooms for a specific time-frame and location.
- **Student events**: A student can access vszhaw news and events.

2.4 Product Backlog

We divided each primary function into seperate smaller user stories to plan our sprints. Additionaly we have defined general user stories that are not directly related to any of the primary functions. All implemented user stories are listed in the following sections. A full list of issues and user stories can also be found as issues on our GitHub repository [9].

2.4.1 General

- **US01**: As a student I want to save my credentials/username
- **US02**: As a student I want the app to work even when I don't have network connection
- **US03**: As a student I want to switch between contexts (schedule, menus, room search, vszhaw)

2.4.2 Timetable

- **US10**: As a student I want to view my timetable for a day
- **US11**: As a student I want to view my timetable for a week
- **US12**: As a student I want to navigate to the current day (timetable)
- **US13**: As a student I want to navigate between days when using the day view (timetable)
- **US14**: As a student I want to navigate between weeks (timetable)
- **US16**: As a student I want to view a specific room's timetable
- **US17**: As a student I want to view a specific class's timetable
- **US18**: As a student I want to view a specific course's timetable
- **US19**: As a student I want to view a specific person's timetable
- **US20**: As a student I want to have a detailed view of my events

2.4.3 Menu plans

- **US50**: As a student I want to view the mensa menu for my campus for a day
- **US51**: As a student I want to view the mensa menu for my campus for a week
- **US52**: As a student I want to navigate to the mensa menu of the current day
- **US53**: As a student I want to navigate between days when using the mensa menu day view
- **US54**: As a student I want to navigate between weeks when using the mensa menu week view
- **US56**: As a student I want to see prices for all menus
- **US57**: As a student I want to navigate between menus of different days
- **US58**: As a student I want to view a specific mensa menu plan

2.4.4 Room search

- **US30**: As a student I want to find currently unoccupied rooms
- **US31**: As a student I want an overview of my campus with highlighted buildings where there are unoccupied rooms
- **US32**: As a student I want a floor plan of each floor per building with highlighted unoccupied rooms
- **US33**: As a student I want to navigate between buildings through the overview of my campus
- **US34**: As a student I want to navigate between floor plans of a building
- **US35**: As a student I want to see until when a room is unoccupied
- **US36**: As a student I want to filter my search to only show rooms that are unoccupied for at least x hours/minutes

2.4.5 Student events

- **US70**: As a student I want to view vszhaw blog posts/event announcements
- **US71**: As a student I want to see upcoming vszhaw events (f. ex. next party)

3 Implementation

3.1 Architecture

The architecture for ZHAWo consists of a frontend React web application [13] and a backend REST server. The server is implemented with the Node.js [14] framework Express [15]. The frontend fetches data such as timetables, mensa menus and event feeds from the backend through a REST API. The backend server - apart from providing the REST API for the frontend - fetches timetable and menu data from the CampusInfo REST API provided by the ZHAW and fetches vszhaw news and calendar events through their RSS feed [2] and their calendar application [3].

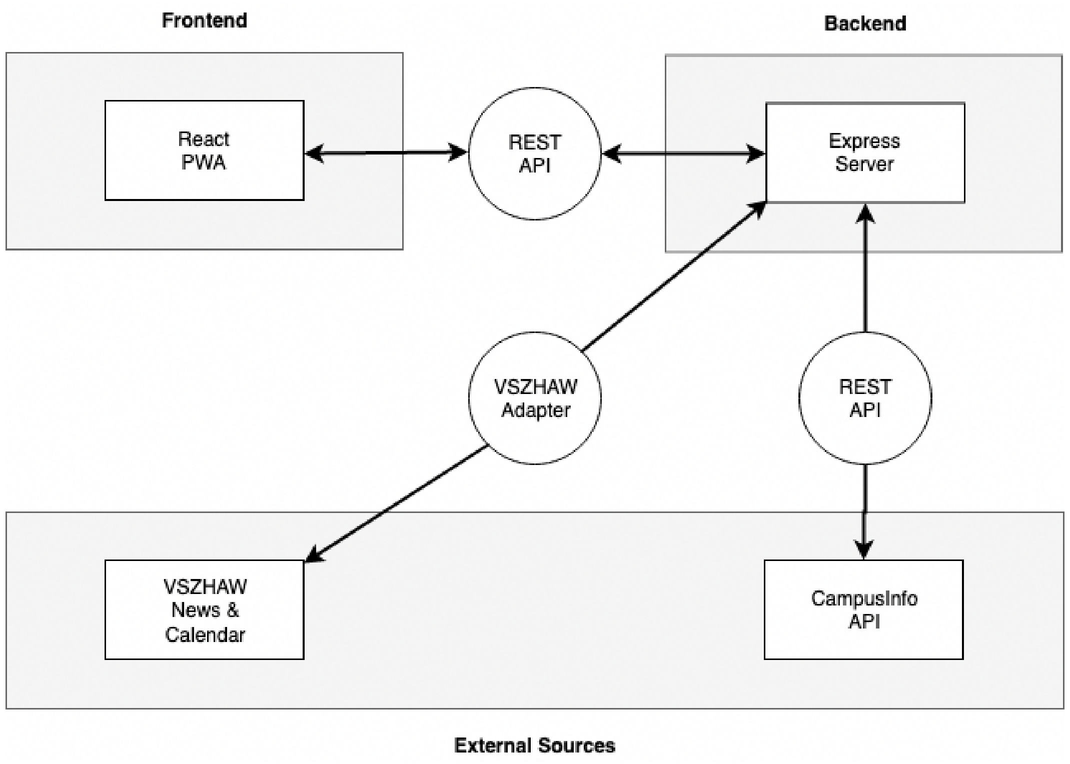


Figure 1: **Application architecture diagram**: React frontend application communicating with backend Express server through a REST API. Express server fetches data from vszhaw RSS feed and calendar web application as well as the CampusInfo REST API for timetable and menu data.

3.2 Backend

The backend architecture consists of the following modules:

- **Express server application**: Handles HTTP requests from frontend and redirection to persistance layer or third-party adapters.
- **Persistence layer**: Handles caching of more resource intensive requests such as room search. Implemented with basic file system using JSON file in the prototype. Can be extended by database if functionality requires.
- **REST API adapter**: Handles HTTP fetch requests to the CampusInfo API.
- **Vszhaw adapter**: Handles fetching of RSS feed and calendar events from vszhaw.

3.2.1 Express Server Application

For the backend server application, we have chosen to use the framework Express [15] based on the Node.js [14] technology. The server offers a REST API to get all the data the frontend needs. Most of the server logic handles redirecting user requests from the frontend to either the persistence layer or the different API adapters and then serve the data. Express is the most used JavaScript web application framework with these capabilities. We chose to implement the backend with a JavaScript based framework so there is no technology difference between front- and backend. In the context of agile development in a small team, this allows for a fast implementation of features. A new feature often requires changes to both front- and backend, and by removing the need to switch context between different programming languages, we were able to maintain a good velocity throughout our sprints.

3.2.2 Persistence Layer

In the scope of this project, there was no need to implement a full database for data persistence. The data for more resource intensive requests such as the search for free rooms is currently stored in JSON format directly in the servers file system. Less intensive requests such as timetables are delivered from CampusInfo in a format that only needs minimal modifications and are directly sent to the frontend without persistence.
However, should the need for a database later arise through new functionality, integration into the Express server application is already prepared.

3.2.3 REST API Adapter

Most of the application data for ZHAWo is provided by the CampusInfo API. Since this is a third-party API provided by the ZHAW, we decided to implement an additional layer between our backend application core and the API. This insures that we are flexible to changes to CampusInfo.

3.2.4 Vszhaw adapter

Similarly to the CampusInfo API adapter, the vszhaw adapter provides an additional layer between our application and the RSS feed [2] of the vszhaw as well as the calendar web application [3]. This ensures that should the data received by this third-party feed and calendar application change, that changes needed to our application will be isolated to the adapter.

3.3 Frontend

The frontend is made up of three main parts:

- **React web application**: Handles presentation of data and user interaction.
- **REST API adapter**: Handles HTTP fetch requests to the backend.
- **Service worker**: Handles PWA functionality such as caching of data for offline use and installation to desktop or phone homescreen.

3.3.1 React Web Application

For the presentation of the web application to the user, we used the React framework [13]. React was originaly developed by Facebook and is one of the most popular UI libraries in web development. It is based on reusable components built with JSX, a syntax extension to JavaScript. We decided to use React because of its component based modularity, which works well with an agile development approach where multiple features need to be implemented simultaneously with little interference.
To handle the application data we chose to use the Flux design pattern [16]. Using the Flux pattern ensures a unidirectional data flow from view components through actions into a single dispatcher into data stores, where the application data such as timetables and menu plans is handled (Figure 2). In Flux, the dispatcher is a singleton that directs the flow of data to ensure that updates do not cascade, which would lead to unpredictable behavior. When a user interacts with a React view, the view sends an action through the dispatcher, which notifies the stores that hold the application's data. When the stores change state, the view gets notified and changes accordingly [16].

3.3.2 REST API Adapter

The data such as timetables, menu plans or lists of free rooms are provided by the backend REST API. To ensure modularity between front- and backend, we implemented an adapter module that handles all data requests to the backend. This practice allows easier adaptations to changes in API, as only the adapter would have to be changed, while the other parts such as the React application and service worker do not need to change.
The modularity this design choice provides - similarly to the modularity of React - goes well with the agile principle of fast prototyping of new ideas and features.

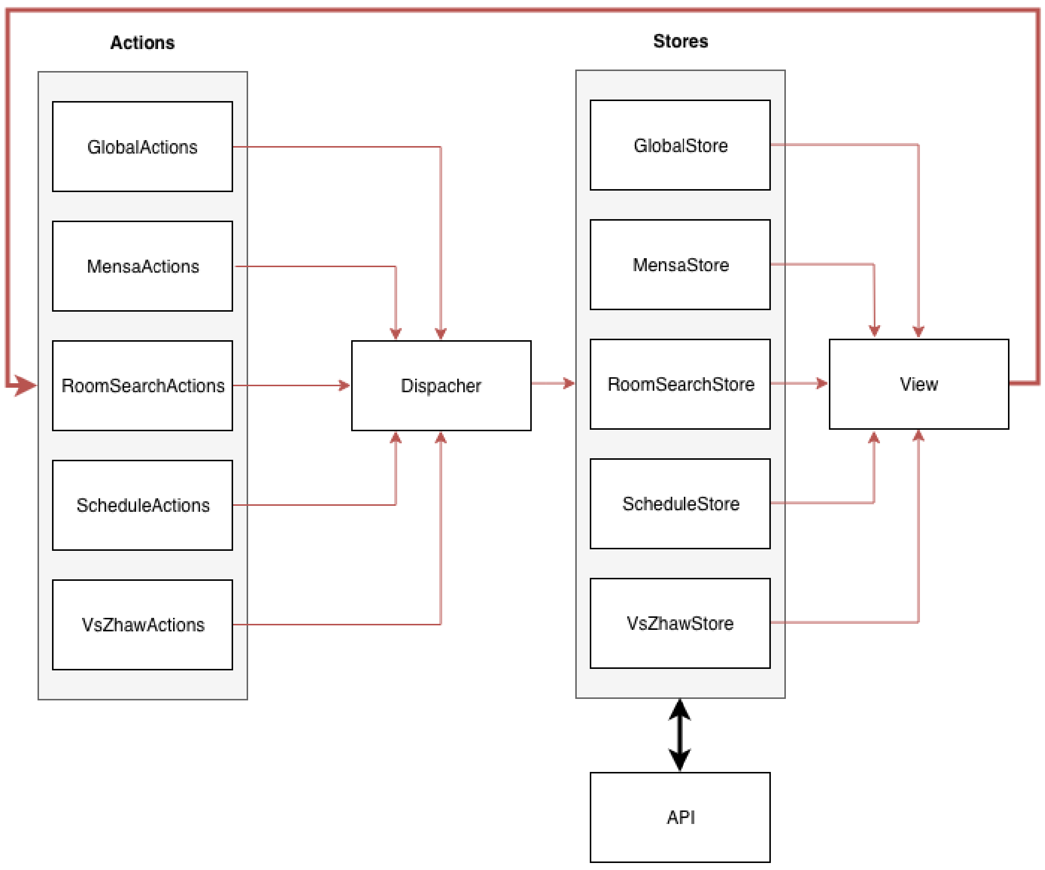


Figure 2: **Flux Pattern diagram**: Unidirectional data flow from view React components through actions to the dispatcher singleton to stores. Views update based on changes in stores.

3.3.3 Service Worker

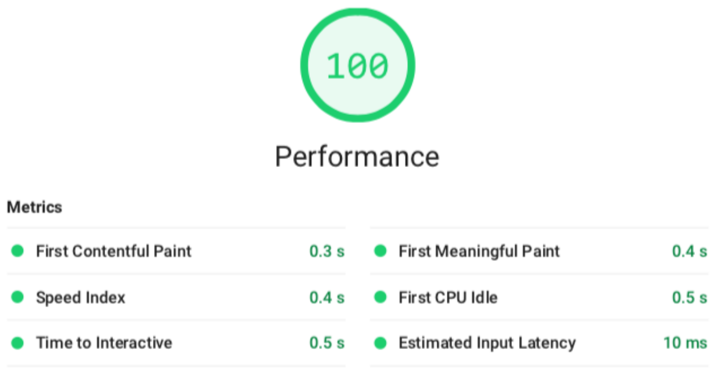
The data for the frontend is provided by the backend REST API. All data received from HTTP fetch requests is cached using a service worker [17]. By caching the application data, we ensure that the user can still access all the information that was already loaded once. It also improves the loading speed of ZHAWo, since resources that have been cached are first served from cache, before the data is requested by the backend. After the fetch request has completed the cache is updated. With regard to, for example, a students schedule information, this practice makes a lot of sense since the actual data does not change often during the course of a semester.
In addition to offline caching, the service worker also allows users to install ZHAWo as a Progressive Web App (PWA) [18] to either their desktop or their mobile phone's home screen.

3.4 Test coverage

To test our application we used the framework Jest [19] for both front- and backend. To test the React components, we used the GUI testing framework [20] in addition to Jest.
For the express server application, we achieved a test coverage of 91%. For the frontend React application 86% of the code was covered, resulting in an overall project test coverage of about 87% [21]. A detailed coverage can be found on our GitHub repository [9] and on our Codecov page [21].
Good testing practices and a high test coverage were important to us. Especially with our agile development approach, this was vital to maintan a high velocity by not introducing unexpected breaking changes between sprints.

3.5 Metrics

Google provides the automated tool Lighthouse [22] to audit and rate PWAs. Besides providing performance metrics, Lighthouse can also be used to test various PWA aspects, including if the application still responds without network connection. ZHAWo achieved a performance rating of 100 out of 100, confirming the user feedback that the app has a very good performance (see Feedback Section 5). ZHAWo also passed all automated PWA checks confirming that it performs well even on slow networks and caching of offline content through the service worker works as intended. Manual tests confirm that as long as a resource such as a timetable or a mensa menu was previously fetched, the application still functions even without any connection to the internet. A detailed report of the Lighthouse audit can be found on the Github Repository [9].
During the last three weeks we published the application to a broader range of students, outside our small testing and feedback group. To date ZHAWo has had about 70 unique users. We were also positively surprised to see that the users were using the application for everything. To date the schedule has been used 411 times, the room search 295 times, the mensa 373 times and vszhaw 186 times. This user base consists of the participants of the survey and people who either heard about the application through those participants. It also consists of our testing group users. We expect the number of individual users to increase at the start of the next semester when we will send out an email about ZHAWo in collaboration with the vszhaw.



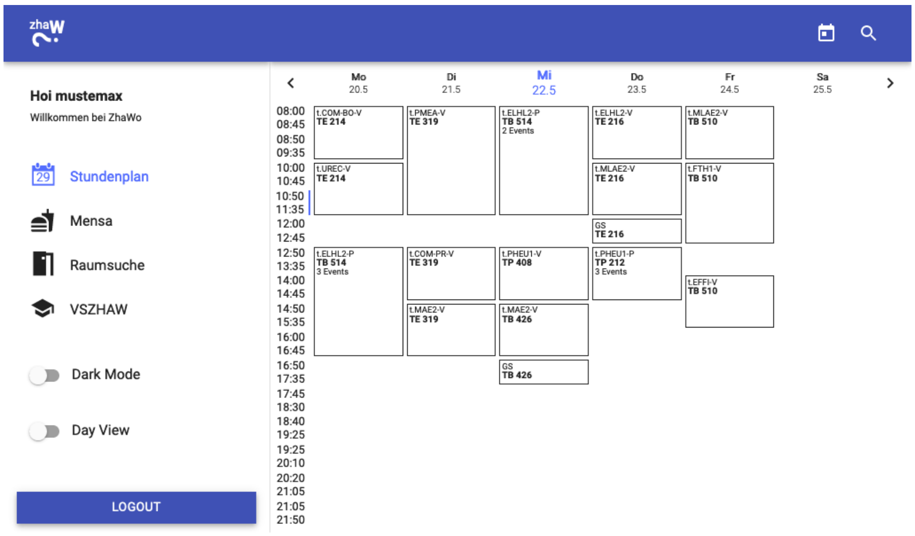
Metrics

● First Contentful Paint	0.3 s	● First Meaningful Paint	0.4 s
● Speed Index	0.4 s	● First CPU Idle	0.5 s
● Time to Interactive	0.5 s	● Estimated Input Latency	10 ms

Figure 3: **Lighthouse performance audit**: ZHAWo was rated by Lighthouse with a performance score of 100 out of 100 points.

4 Application

The user interface was designed to provide students with the familiar look and feel of a native mobile application. We chose a minimalistic design approach. The focus was on displaying relevant information without any distracting noise or clutter and a fast and intuitive user experience. Users can choose between a light theme and a dark theme (Figures 6, 9 and 11). On desktop devices the drawer is always open and contains the navigation (Figures 4 and 5). A detailed interaction flow with ZHAWo is described in the following sections.



	Mo 20.5	Di 21.5	Mi 22.5	Do 23.5	Fr 24.5	Sa 25.5
08:00 08:45	t.COM-BO-V **TE 214**	t.PMEA-V **TE 319**	t.ELHL2-P **TB 514** 2 Events	t.ELHL2-V **TE 216**	t.MLAE2-V **TB 510**	
08:50 09:35						
10:00 10:45	t.UREC-V **TE 214**			t.MLAE2-V **TE 216**	t.FTH1-V **TB 510**	
10:50 11:35						
12:00 12:45				GS **TE 216**		
12:50 13:35	t.ELHL2-P **TB 514** 3 Events	t.COM-PR-V **TE 319**	t.PHEU1-V **TP 408**	t.PHEU1-P **TP 212** 3 Events		
14:00 14:45					t.EFFI-V **TB 510**	
14:50 15:35		t.MAE2-V **TE 319**	t.MAE2-V **TB 426**			
16:00 16:45						
16:50 17:35			GS **TB 426**			
17:45 18:30						
18:40 19:25						
19:25 20:10						
20:20 21:05						
21:05 21:50						

Figure 4: **Desktop timetable user interface**: Desktop interface of a student's timetable in week view with expandend drawer.

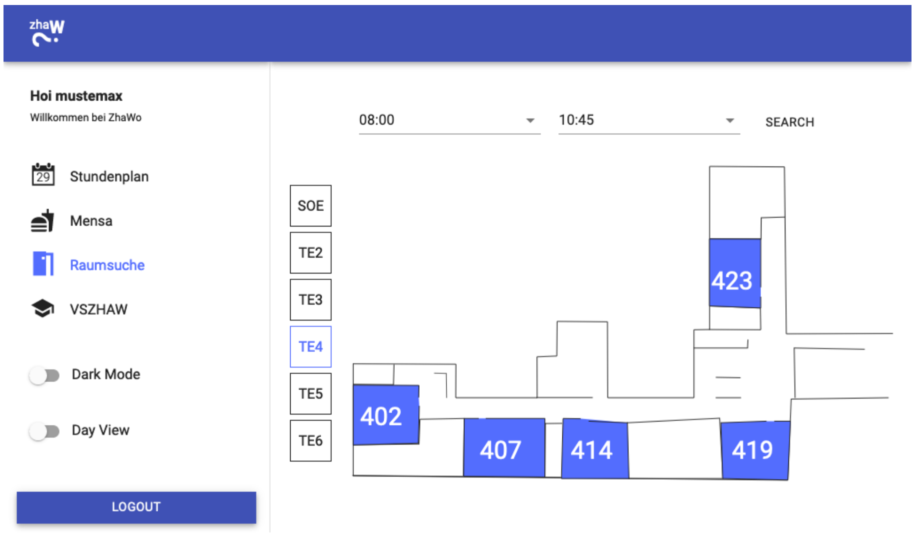


Figure 5: **Desktop room search user interface**: Desktop interface of a room search showing free rooms on the floor plan of the fourth floor of the TE building.

4.1 General

On a users first visit to ZHAWo, a prompt is shown to enter their ZHAW username (Figure 6 A). A list of all available student and professor names is loaded and on user input, suggestions from that list are displayed. Since there is no persistent user specific data, setting up an account is not needed. The ZHAW username is required to fetch the correct timetable. This username is stored and on the next visit, the initial screen is skipped and users have immediate access to their timetable.

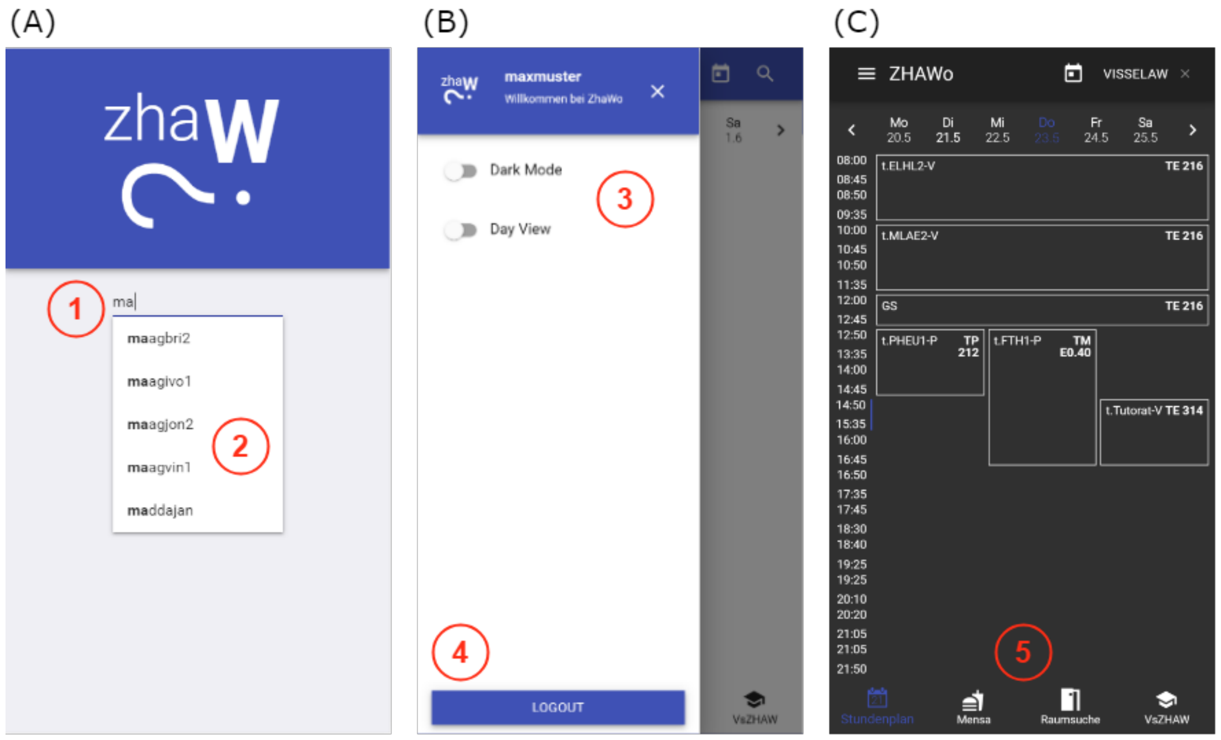


Figure 6: **General user interface**: **A**: On first visit, users are prompted to enter (1) and select (2) their ZHAW username. **B**: Through a collapsible application drawer, users have access to application settings (3) to change the application theme and timetable display mode. Users can clear their stored ZHAW username through a logout button (4). **C**: Users can switch between timetable, mensa menu, room search and student events contexts through labeled icons on a bottom navigation bar (5).

Through a collapsible application drawer, users have access to some basic settings like switching between display modes of the timetable and changing the application theme (Figure 6 B). The two timetable display modes show the user's timetable for either a single day or a full week. The saved ZHAW username can be cleared by clicking the logout button at the bottom of the application drawer.
Switching between contexts (corresponding to the four primary functions timetable, mensa menus, room search and student events) is done through labeled icons on a navigation bar at the bottom of the screen (Figure 6 C). The application state such as selected date or time frame for the room search are preserved between context switches.

4.2 Timetable

By default, the timetable is displayed in a day view (Figure 7 A). Individual events are displayed in their corresponding time slots. The time slots are displayed on the left with a subtle indicator for the current time slot. If there are events that overlap in time, they are displayed next to each other in the day view. To navigate between different dates, users can swipe to the right to advance one day or swipe to the left to go to the previous day respectively. Above the timetable, a navigation bar is displayed which highlights the currently displayed day or week. As an alternative to the swipe interactions, users can directly select a date in the navigation bar or jump one week back or forward by clicking on the arrow buttons. To quickly jump back to the timetable of day current day or week, users can click on the calendar icon in a context action area on the top right of the screen (Figure 7 B).
In addition to the day view, users can choose to display their timetable for a whole week. Since display space is limited especially on small mobile devices, overlapping events are shown as an indicator below the longest running event during that time frame as seen in Figure 7 B.
When clicking on a specific event, a modal pops up displaying detailed information about that event (Figure 7 C).

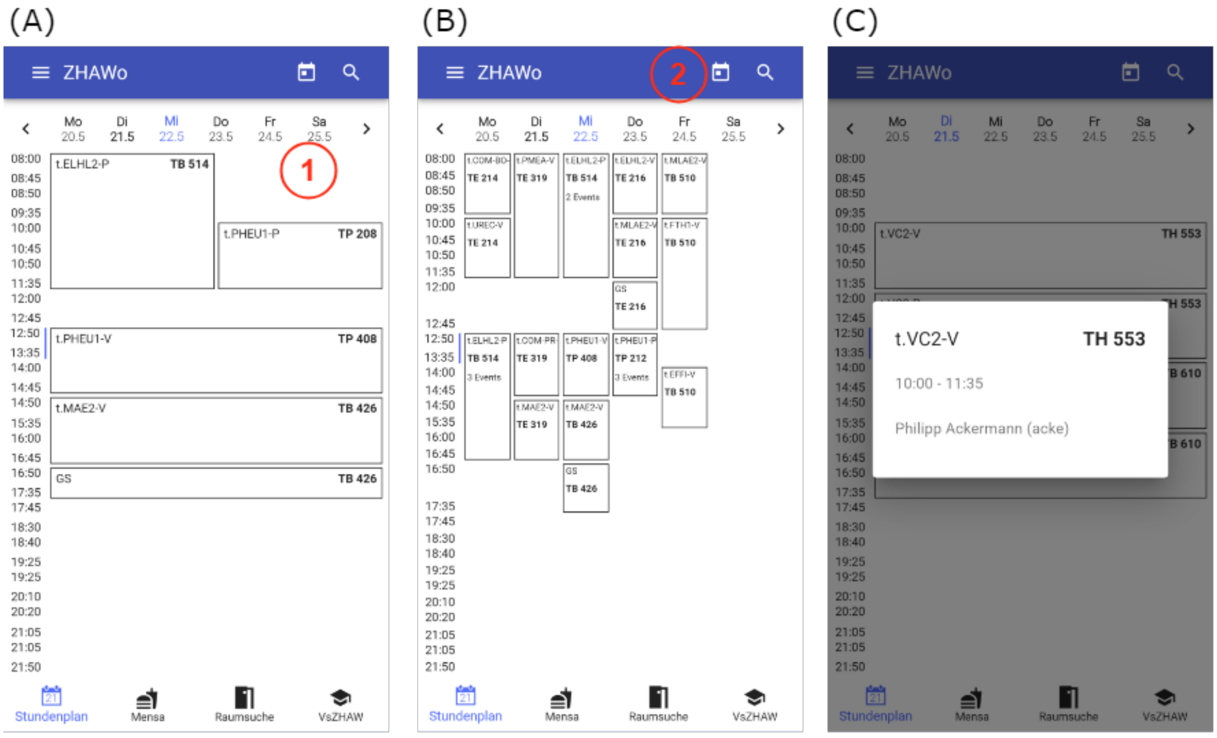


Figure 7: **Timetable user interface**: **A**: Day view of a users timetable with overlapping events from 10:00 to 11:35. Navigation bar to navigate to specific dates and to navigate between weeks (1). Alternatively, navigation between days or weeks can be achieved through swipe gestures. **B**: Week view of a users timetable with overlapping event indicators and context action area button to jump to current date (2). **C**: Detail modal of a specific event. Is opened when user clicks on event from either day or week view.

Users can also search for and display other timetables than their own. Clicking on the search icon button in the context action area opens a modal. Users can search for and display timetables of other students, lecturers, classes, courses or rooms (Figure 8 A and B). The currently displayed search timetable can be cleared by clicking the indicator in the context action area (Figure 8 C).

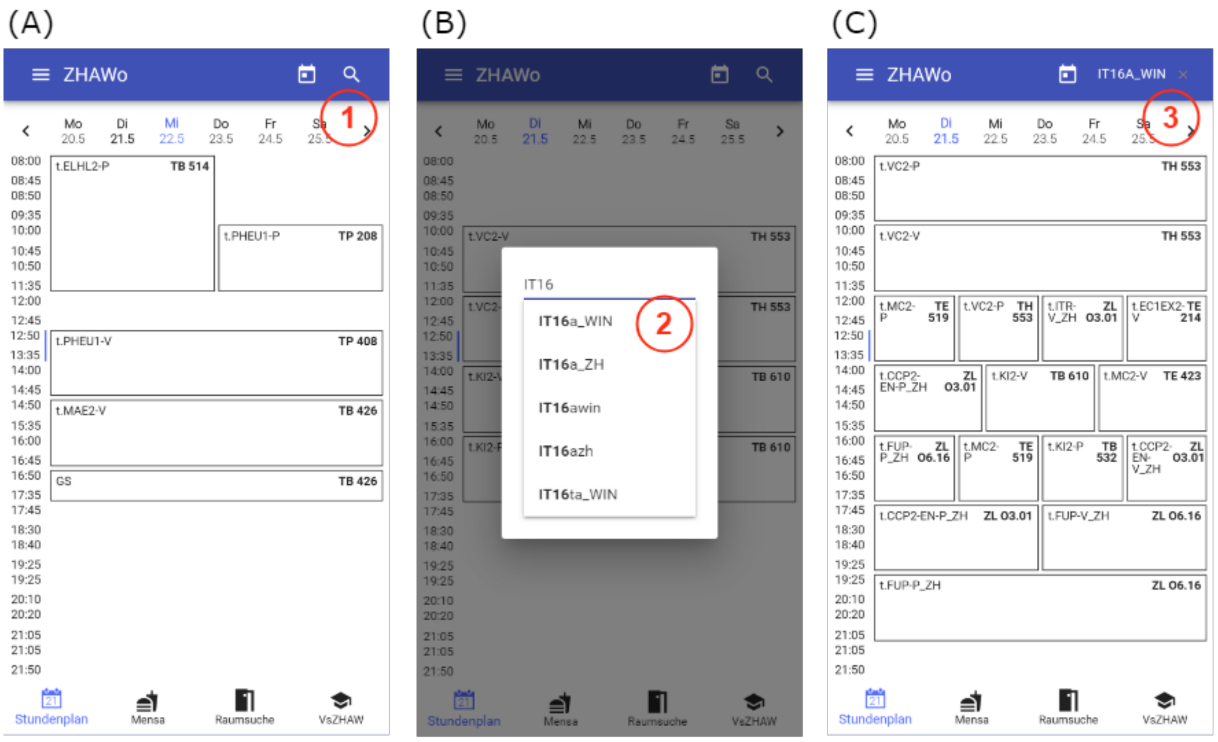
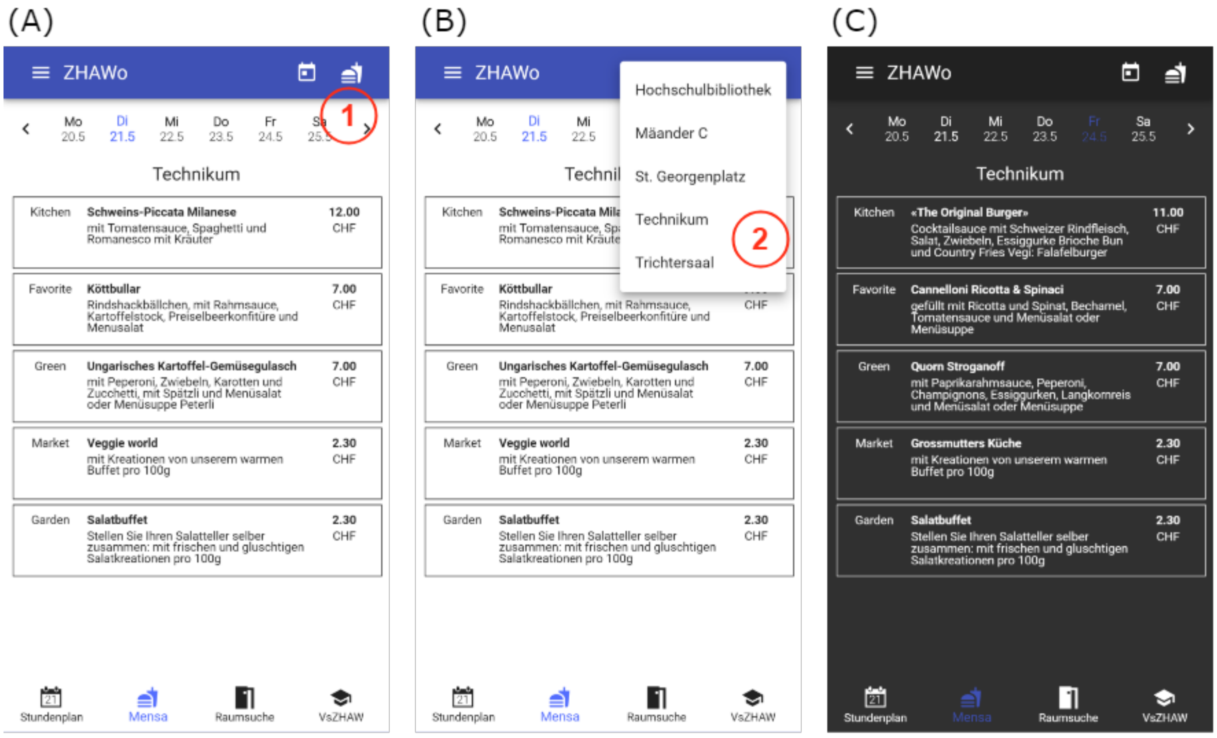


Figure 8: **Timetable search user interface**: **A**: Default user timetable with search icon button in context action area to open search modal (1). **B**: Search modal with suggestions for students, lecturers, classes, courses and rooms (2). **C**: Search timetable for specific class with indicator and clear button (3) in context action area.

4.3 Mensa menus

The mensa menus are displayed as a list of available menus with their category, menu name and description and ZHAW internal prices. The title indicates the mensa where the currently displayed menu is offered. By clicking the mensa icon in the context action area, users can switch between different mensas (Figure 9 A and B). Navigation between menus of different days is consistent with the navigation for the timetable context, with a navigation bar in addition to swipe gestures.



Technikum

Kitchen	**Schweins-Piccata Milanese** mit Tomatensauce, Spaghetti und Romanesco mit Kräuter	12.00 CHF
Favorite	**Köttbullar** Rindshackbällchen, mit Rahmsauce, Kartoffelstock, Preiselbeerkonfitüre und Menusalat	7.00 CHF
Green	**Ungarisches Kartoffel-Gemüsegulasch** mit Peperoni, Zwiebeln, Karotten und Zucchetti, mit Spätzli und Menüsalat oder Menüsuppe Peterli	7.00 CHF
Market	**Veggie world** mit Kreationen von unserem warmen Buffet pro 100g	2.30 CHF
Garden	**Salatbuffet** Stellen Sie Ihren Salatteller selber zusammen: mit frischen und gluschtigen Salatkreationen pro 100g	2.30 CHF

Technikum

Kitchen	**«The Original Burger»** Cocktailsauce mit Schweizer Rindfleisch, Salat, Zwiebeln, Essiggurke Brioche Bun und Country Fries Vegi: Falafelburger	11.00 CHF
Favorite	**Cannelloni Ricotta & Spinaci** gefüllt mit Ricotta und Spinat, Bechamel, Tomatensauce und Menüsalat oder Menüsuppe	7.00 CHF
Green	**Quorn Stroganoff** mit Paprikarahmsauce, Peperoni, Champignons, Essiggurken, Langkornreis und Menüsalat oder Menüsuppe	7.00 CHF
Market	**Grossmutters Küche** mit Kreationen von unserem warmen Buffet pro 100g	2.30 CHF
Garden	**Salatbuffet** Stellen Sie Ihren Salatteller selber zusammen: mit frischen und gluschtigen Salatkreationen pro 100g	2.30 CHF

Figure 9: **Mensa menu user interface**: **A**: Mensa menu for the Technikum mensa with menu category, name, description and ZHAW internal prices. Title indicates currently displayed mensa which can be changed through the mensa icon in the context action area (1). **B**: Mensa selection (2). **C**: Dark theme mensa menu display.

4.4 Room search

To search for currently unoccupied rooms at the ZHAW School of Engineering Technikum campus, users can enter a time range during which they want to use the room (Figure 10 A). Once they've entered the start and end time, they can press the search button and the rooms will be filtered to only show rooms that are unoccupied during the chosen time slots. To browse the free rooms, the users can navigate through a map of the campus showing all the buildings. Buildings highlighted in blue have at least one free room during the entered time range (Figure 10 B). To see which rooms are free, users can click on a building on the map, and a floor plan of that building will appear with labeled and highlighted free rooms. They can navigate through the different floors with the buttons above the floor plans. The individual floor plans show which rooms are free and where they are (Figure 10 C). To switch to a different building, users can navigate back to the campus overview map through the School Of Engineering button next to the floor navigation.

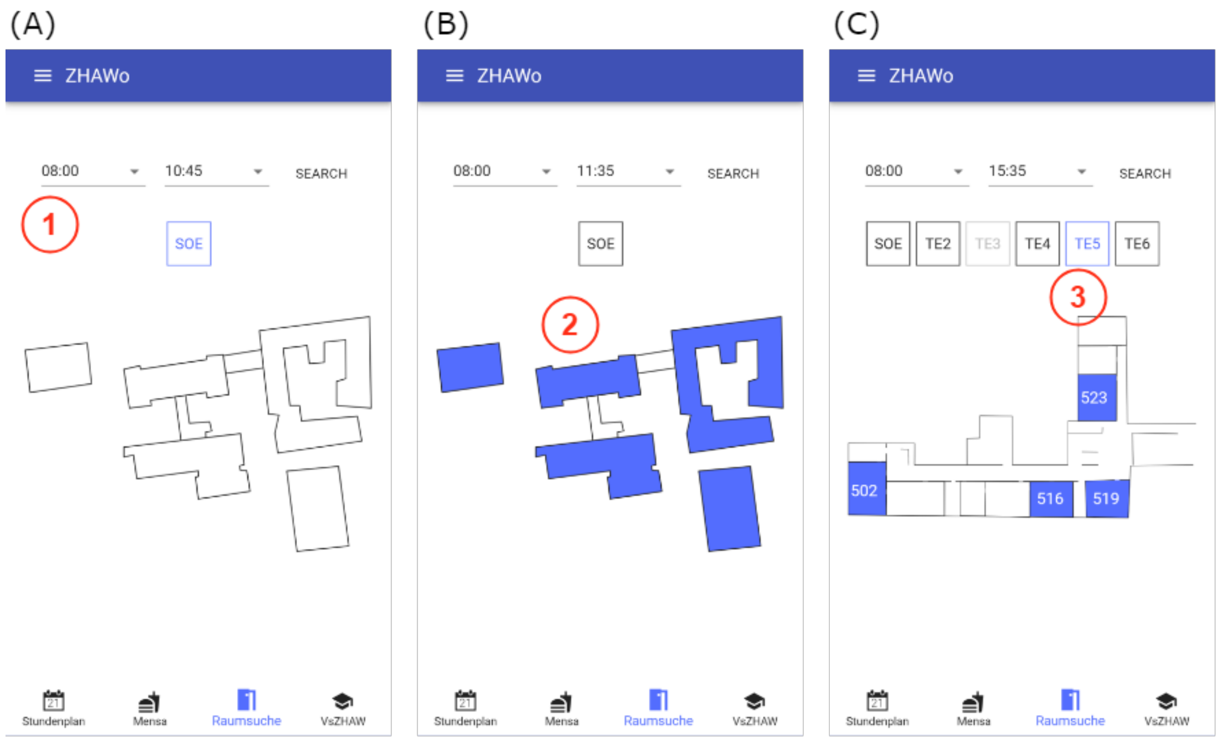
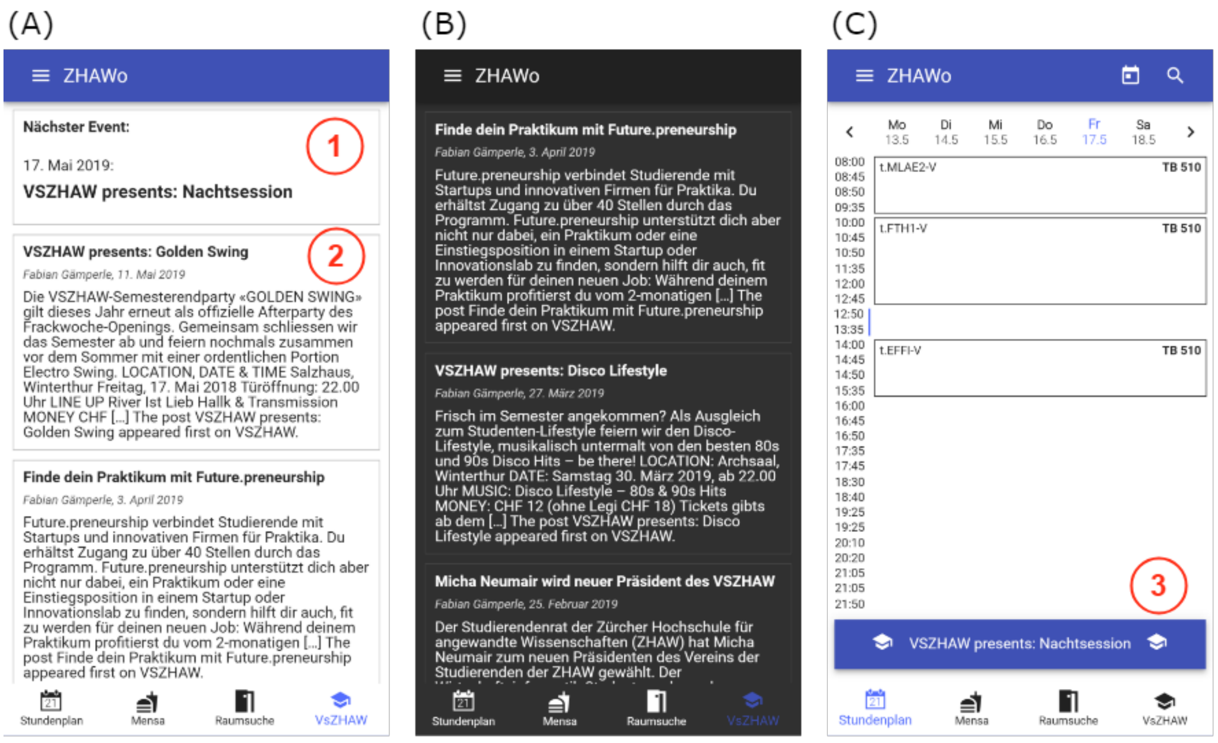


Figure 10: **Room search user interface**: **A**: Initial screen of the room search without an active search with input for start and end time and search button (1). **B**: Campus overview map of the School Of Engineering with buildings highlighted in blue that contain a free room in the entered time range. By clicking on a building on the map, users get directed to the respective floor plans of that building (2). **C**: Detailed floor plan of TE building with free rooms highlighted and labeled. Navigation through different floors with buttons above the map (3).

4.5 Student events

Users can browse vszhaw news as a chronological list of blog posts. The full news posts are linked and can be reached by clicking on the individual posts. If there is an upcoming event in the official vszhaw calendar, it will be featured above the news feed (Figure 11 A and B). Additionally, student events are displayed as a banner in the timetable context of their date (Figure 11 C). Both the banner and student event feature are linked to more a detailed view of the event on the vszhaw website.



Nächster Event:

17. Mai 2019:

VSZHAW presents: Nachtsession

VSZHAW presents: Golden Swing

Fabian Gämperle, 11. Mai 2019

Die VSZHAW-Semesterendparty «GOLDEN SWING» gilt dieses Jahr erneut als offizielle Afterparty des Frackwoche-Openings. Gemeinsam schliessen wir das Semester ab und feiern nochmals zusammen vor dem Sommer mit einer ordentlichen Portion Electro Swing. LOCATION, DATE & TIME Salzhaus, Winterthur Freitag, 17. Mai 2018 Türöffnung: 22.00 Uhr LINE UP River Ist Lieb Hallk & Transmission MONEY CHF [...] The post VSZHAW presents: Golden Swing appeared first on VSZHAW.

Finde dein Praktikum mit Future.preneurship

Fabian Gämperle, 3. April 2019

Future.preneurship verbindet Studierende mit Startups und innovativen Firmen für Praktika. Du erhältst Zugang zu über 40 Stellen durch das Programm. Future.preneurship unterstützt dich aber nicht nur dabei, ein Praktikum oder eine Einstiegsposition in einem Startup oder Innovationslab zu finden, sondern hilft dir auch, fit zu werden für deinen neuen Job: Während deinem Praktikum profitierst du vom 2-monatigen [...] The post Finde dein Praktikum mit Future.preneurship appeared first on VSZHAW.

Finde dein Praktikum mit Future.preneurship

Fabian Gämperle, 3. April 2019

Future.preneurship verbindet Studierende mit Startups und innovativen Firmen für Praktika. Du erhältst Zugang zu über 40 Stellen durch das Programm. Future.preneurship unterstützt dich aber nicht nur dabei, ein Praktikum oder eine Einstiegsposition in einem Startup oder Innovationslab zu finden, sondern hilft dir auch, fit zu werden für deinen neuen Job: Während deinem Praktikum profitierst du vom 2-monatigen [...] The post Finde dein Praktikum mit Future.preneurship appeared first on VSZHAW.

VSZHAW presents: Disco Lifestyle

Fabian Gämperle, 27. März 2019

Frisch im Semester angekommen? Als Ausgleich zum Studenten-Lifestyle feiern wir den Disco-Lifestyle, musikalisch untermalt von den besten 80s und 90s Disco Hits – be there! LOCATION: Archsaal, Winterthur DATE: Samstag 30. März 2019, ab 22.00 Uhr MUSIC: Disco Lifestyle – 80s & 90s Hits MONEY: CHF 12 (ohne Legi CHF 18) Tickets gibts ab dem [...] The post VSZHAW presents: Disco Lifestyle appeared first on VSZHAW.

Micha Neumair wird neuer Präsident des VSZHAW

Fabian Gämperle, 25. Februar 2019

Der Studierendenrat der Zürcher Hochschule für angewandte Wissenschaften (ZHAW) hat Micha Neumair zum neuen Präsidenten des Vereins der Studierenden der ZHAW gewählt. Der

Figure 11: **Student events user interface**: **A**: Vszhaw student events (1) and news feed (2). **B**: Dark theme version of vszhaw news feed. **C**: Student event banner in the timetable context (3).

5 Feedback

To distribute our application and receive user feedback, we demonstrated our application to random students on the School Of Engineering Technikum campus. We asked the students to use our application for a bit and then fill out a survey. The survey contained questions about the usefulness, performance and design of ZHAWo as well as open questions about general feedback and improvement ideas. Out of 37 total survey participants, 76% (28) rated the usefulness of ZHAWo with 5 out of 5 points (Figure 12) and the average rating of the usefulness with 4.75 out of 5 was very high. The performance was rated with an average of 4.62 out of 5 points (Figure 13). The design of the application was rated with a lower point score of on average 4.24 out of 5 points (Figure 14). The design rating is rather subjective, with some participants pointing out that they much prefer the simplistic design of ZHAWo over the official applications, while others list the design as an area where ZHAWo could improve. Some of the lower ratings for the design can be attributed to the fact that especially on Huawei phones, the floor plans were not always displayed correctly.
In general, the room search feature and it's implementation with floor plans that display the locations of free rooms was received very positively, with most participants pointing out that feature as very useful and something they had been wishing for. A few participants also asked if the room search could be expanded to allow them to search for free rooms on a specific date. Others said that it should be expanded to also include other locations and not just the School Of Engineering campus at the Technikum.
The way we handled the display of overlapping events for timetables was also well received and was noted to be more readable compared to the official CampusInfo app.
In addition to getting feedback on our application, we also wanted to get a sense of how familiar people are with the concept of Progressive Web Applications. A majority of the survey participants (78%) have not previously heard about PWAs. This coincided with what expected. PWAs are a new technology and people are only just starting to learn about them.
Detailed results of the survey can be found on our GitHub repository [9].

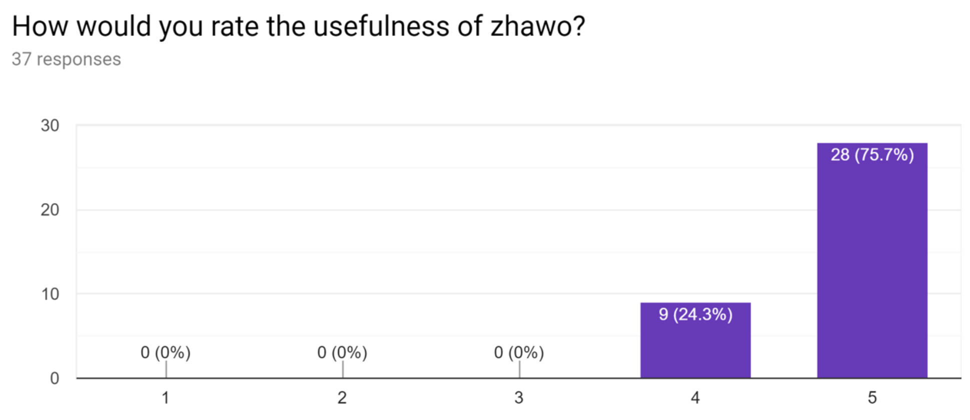


Figure 12: **Usefulness**: Distribution of the ratings for the usefulness of ZHAWo.

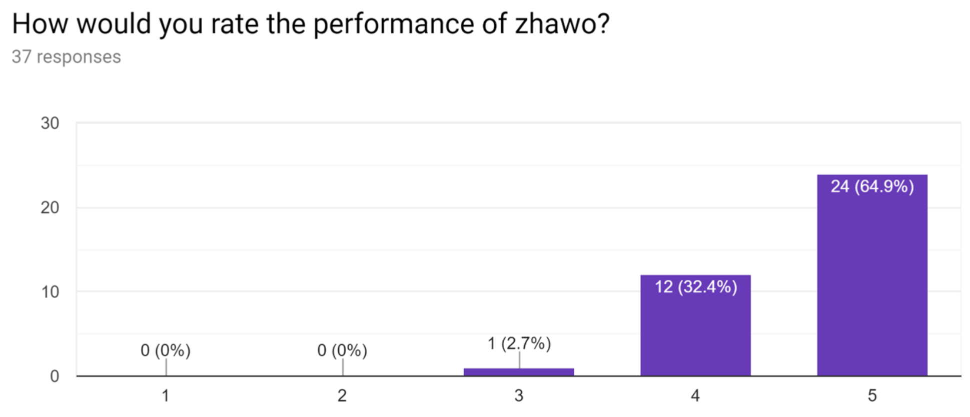


Figure 13: **Performance**: Distribution of the ratings for the performance of ZHAWo.

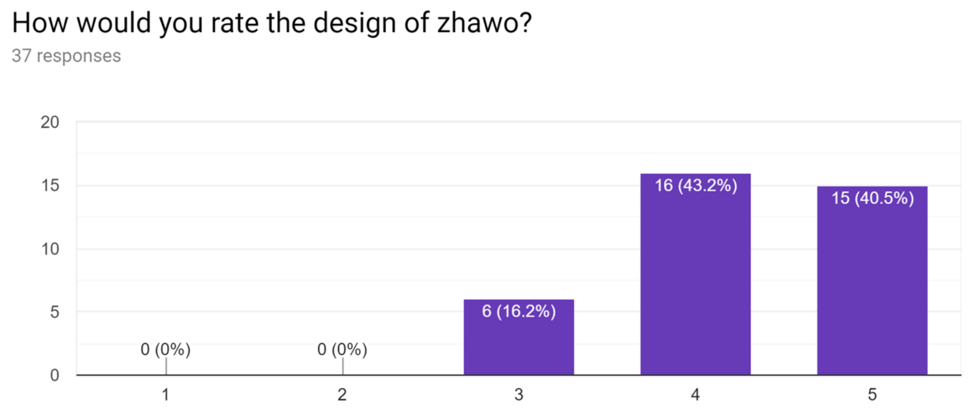


Figure 14: **Design**: Distribution of the ratings for the design of ZHAWo.

6 Discussion

6.1 ZHAWo

We are happy with the outcome of ZHAWo. The feedback we received was also very positive. The user survey confirmed that a room search feature is in high demand amongst students of the ZHAW. Our implementation of the building and floor plans was well received and definitely worth the effort of drawing each plan by hand. With the updates to the official CampusInfo application for iOS, a good solution to view timetable and mensa menus was less important than at the start of this project, however to us it makes sense to combine these feature and the room search functionality in one single application. In collaboration with the vszhaw we hope to gain new users at the begin of the next semester.

6.2 PWA

Using a full JavaScript technology stack for front- and backend and Progressive Web Apps has proven to work well with an agile development approach. While developing native applications, having to maintain different code bases can be awkward in combination with agile principles. The strength of the agile development approach that we have chosen is that we can prototype new features rapidly. This allows us to receive user feedback from both Android and iOS users immediately. In addition to that, not having to stay up to date with best practices, libraries and frameworks for different languages such as Java/Kotlin and Swift/Objective C for Android and iOS development respectively allowed us to implement new features at a faster pace.
While there are obvious advantages to building ZHAWo as a cross platform PWA, we also identified some weaknesses. Our user survey indicated that installing an application from within the browser and not through the App Store or the Google Play Store is unexpected for most users. There are also inconsistencies in support offered by the different platforms, similar to the browser inconsistencies in web development [23].
We identified additional issues and inconsistencies in rendering of our application on different smartphone browsers. Especially the scaling of the SVG images of the building and floor plans was not optimal on some smaller phones and specifically on Huawei smartphones. These issues are not specific to PWA technology, but arise for all mobile websites.
Building PWAs gives you the the benefits of web development, for instance being platform independent. But it also comes with the problems of web development. Because the application relies on a browser to be displayed. You don't have control over how that browser is implemented and how it displays your application. This causes issues and should be considered when deciding between using native technologies or a PWA.
Some of our issues can be attributed to PWA technologies being relatively new. We expect both support and users familiarity with the concept of PWAs to increase in the near future. At the moment the world of PWAs is a bit chaotic. The different smartphone and browser manufactures do not agree on what a PWA is. Many of them are constantly changing their implementation and support of the technology.
Considering both advantages and disadvantages of PWA technology in it's current state, we would currently not recommend building a PWA as a primary application. PWAs are definitely an interesting concept and we look forward to seeing what the future holds for them.

7 Appendix

List of Figures

References

[1] Dominik Bachmann, Julian Visser. (2019). ZHAWo, [Online]. Available: https://zhawo.ml (visited on 05/27/2019).

[2] vszhaw. (2019). vszhaw News Feed, [Online]. Available: https://www.vszhaw.ch/ (visited on 05/27/2019).

[3] ——, (2019). vszhaw Calendar, [Online]. Available: https://www.vszhaw.ch/events/ (visited on 05/27/2019).

[4] ZHAW. (2019). ZHAW Stundenplan, [Online]. Available: http://stundenplan.zhaw.ch (visited on 05/27/2019).

[5] ——, (2019). ZHAW Engineering CampusInfo Android App, [Online]. Available: https://play.google.com/store/apps/details?id=ch.zhaw.init.android.campusinfo (visited on 05/27/2019).

[6] ——, (2019). ZHAW Engineering CampusInfo iOS App, [Online]. Available: https://itunes.apple.com/ch/app/zhaw-engineering-campusinfo/id715684381 (visited on 05/27/2019).

[7] SV Group. (2019). SV Group Menuplan, [Online]. Available: http://technikum.sv-restaurant.ch/de/menuplan/ (visited on 05/27/2019).

[8] Google. (2019). Progressive Web Apps, [Online]. Available: https://developers.google.com/web/progressive-web-apps/ (visited on 05/27/2019).

[9] Dominik Bachmann, Julian Visser. (2019). ZHAWo Github repository, [Online]. Available: https://github.com/zhaw-timetable/zhawo (visited on 05/27/2019).

[10] ——, (2019). ZHAWo Github Wiki, [Online]. Available: https://github.com/zhaw-timetable/zhawo/wiki (visited on 05/27/2019).

[11] travis ci.org. (2019). Travis CI, [Online]. Available: https://travis-ci.org/ (visited on 05/27/2019).

[12] codecov.io. (2019). Codecov, [Online]. Available: https://codecov.io/ (visited on 05/27/2019).

[13] reactjs.org. (2019). ReactJS, [Online]. Available: https://reactjs.org (visited on 05/27/2019).

[14] nodejs.org. (2019). NodeJS, [Online]. Available: https://nodejs.org (visited on 05/27/2019).

[15] expressjs.com. (2019). ExpressJS, [Online]. Available: https://expressjs.com/ (visited on 05/27/2019).

[16] Facebook. (2014). Flux, [Online]. Available: http://facebook.github.io/flux/ (visited on 05/27/2019).

[17] M. Gaunt. (2019). Service Workers: an Introduction, [Online]. Available: https://developers.google.com/web/fundamentals/primers/service-workers/ (visited on 05/27/2019).

[18] P. LePage. (2019). PWA Code Lab, [Online]. Available: https://developers.google.com/web/fundamentals/codelabs/your-first-pwapp/ (visited on 05/27/2019).

[19] Facebook. (2019). Jest, [Online]. Available: `https://jestjs.io/` (visited on 05/27/2019).

[20] Airbnb. (2019). Enzyme, [Online]. Available: `https://airbnb.io/enzyme/` (visited on 05/27/2019).

[21] Dominik Bachmann, Julian Visser. (2019). ZHAWo Codecov project, [Online]. Available: `https://codecov.io/gh/zhaw-timetable/zhawo/` (visited on 05/27/2019).

[22] Google. (2019). Lighthouse, [Online]. Available: `https://developers.google.com/web/tools/lighthouse/` (visited on 05/27/2019).

[23] J. Freestone. (2019). The current state of progressive web apps, [Online]. Available: `https://www.browserlondon.com/blog/2019/04/15/current-state-progressive-web-app-pwa/` (visited on 05/27/2019).