Vietnamese-German University

Technical Report

java group 15

java object oriented programming

Group 15 - Time Scheduler

Authors:
Tim Görß - 1252200
Ante Maric - 1273904
Minh Triet Huynh - 1370690
Jorge Vanegas Aristizabal - 1333459

Submitted to:
PhD. Luigi La Blunda

November 1, 2022

Contents

Chapter 1

Project Description and Motivation

Nowadays, time management apps such as Outlook, Google Calendar,etc have proven to be an essential tool in our modern day business and personal lives.

However, the problem with most of these apps is that they require you to have a constant internet connection to use their application. In addition, the data that you will generate while using these apps belongs to big tech corporations with no clear or transparent way to export it or to view them outside the app once they are created. These factors along with the fact that the majority of these apps are proprietary products mean that they are not an ideal choice for privacy oriented users.

Data protection is a key aspect of our application design. This is why the user's data is backed up in our remote database only if they choose to do so by logging in with an existing account or if they sign up for a new account. Otherwise, if they do not want their data to be stored in our local database, they can choose to use the offline mode, where all of their data is stored and used locally on their computer. Both methods have their own way to transfer the data onto another machine allowing the user to be more flexible.

The main page of the app was designed to offer a modern up-to-date UI that suits the current standards of UI design. Each events and dates are color coded in vibrant colors based on their priority so that color blind users can still interact with the app.

Users are able to create, schedule, edit and delete events, and furthermore invite other users to their event. In addition, our app contains an email function where the users will be reminded to their upcoming event based on the timer option they chose as well as receive emails on upcoming changes or cancellation of event(s) that they are involved in.

What is yours will always rightfully belong to you!

Chapter 2

Project Requirements

Name	Description	Requirement Type	Actors	Member
User sign up	User are able to create an account with an e-mail and password that is backed up in our remote database	functional	User, Admin	Triet Huynh
User login	User can use their associated e-mail and password to log in to their account	functional	User, Admin	Triet Huynh
Change User information	Users can change their e-mail and password	non-functional	User	Jorge
User Setting GUI	Users can change settings using a GUI	non-functional	User	Jorge
User Database	User data is stored in a database	functional	User, Admin	Triet Huynh
E-mail checker	Check if users enter a valid e-mail address	non-functional	User, Admin	Triet Huynh
Password encryption	Password is stored in our database using a encryption method	non-functional	User, Admin	Triet Huynh
Login/Registration GUI	User can register/login using a GUI	functional	User,Admin	Ante, Triet Huynh
Admin access	Admins can login to our software	functionall	Admin	Ante, Triet Huynh
Admin access to user profiles	Admins can see all of the user profiles	functional	Admin	Ante
Admin can edit user profiles	Admins can edit user information such as e-mail and password	functional	Admin	Ante
Admin can delete user	Admins can delete user from database	functional	Admin	Ante
Admin GUI	Admin page GUI	functional	Admin	Ante
Create events	Users are able to create events with specific information	functional	User	Triet Huynh
Event Database	Events are stored in a local database	functional	User	Triet Huynh
Edit events	Users can change event information	functional	User	Tim
Delete events	Users can delete events	functional	User	Tim
E-mail notification when changing events	Users that are participants of an event receive an e-mail when changes were made to an event	non-functional	User	Ante, Tim
Reminder notifications	Users that are participants of an event receive an e-mail when the reminder is triggered	non-functional	User	Ante, Tim
Event creation GUI	Users can use a GUI to create and edit events	functional	User	Tim, Triet Huynh
Calendar View	A interact able calendar inside our software to better visualize time	functional	User	Tim
Events visualized in Calendar	Events are visualized on a calendar and interact able	functional	User	Tim
Events highlighted by priority	Events are highlighted in different colours on our calendar by priority	functional	User	Tim
Event management GUI	Main GUI used for our software	functional	User	Jorge, Tim
Upcoming events	Upcoming events for the month shown on our GUI	non-functional	User	Jorge, Tim
Export schedule as plain text file	Users can export their schedule to a txt file with their local database	non-functional	User	Triet Huynh
Offline Mode	Scheduler is usable without an account	non-functional	User	Triet Huynh
Contact List	Participants' emails are stored in the local database	functional	User	Triet Huynh, Jorge
Sync databases	Users can synchronize their local database with the remote database	non-functional	User	Triet Huynh

Table 2.1: Project Requirements and Task distribution table

Chapter 3

Gantt chart

Tasks	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7
Setting up workplace (IDE, Github, Maven, Overleaf)	🟧						
Drawing diagrams (class diagram, use case diagram)	🟧						
First draft Overleaf documentation (project motivation & description)	🟧						
Creating a database for the user and events		🟥					
Implementing login and sign up function (terminal only)		🟥					
Encryption method for passwords		🟥					
Create GUI for login and sign up			🟫				
Create event function (terminal only)			🟫				
Create GUI for events				🟫			
Create Admin GUI and implement delete/edit functions				🟫			
Implement calendar functions and Jdate picker				🟫			
Implement local database and feature to export database					🟩		
Add participants to events and to friendlist for future events					🟩		
Create email and reminder function						🟪	
Implement change password for users and its GUI						🟪	
Finish working on the documentation							🟦
Review everything and fix bugs if needed							🟦

Table 3.1: Gantt Chart

Chapter 4

Software Description

4.1 GUI

Initially developed using default java swing, we have decided to use a dependency called *FlatLaf* that allows our application to have aesthetics that is up to date with other current applications' standards, since the look of the default swing framework is rather outdated.

4.2 User Registration

4.2.1 Login Page

The login page is the first page that the user will encounter when they start the app. The page is capable of displaying error messages such as: empty email or password field, incorrect email format by using the regex function to check, and finally display an error message indicating whether the information entered is correct or valid for the user to be logged in.

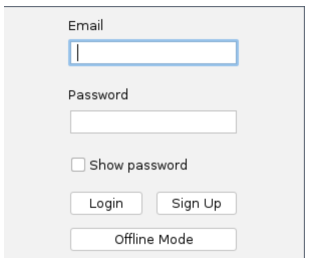

Figure 4.1: Login page image

Two accounts exist on the remote database that are used purely for testing:

- Normal user account email: t@g.com, password: t
- Admin user account email: f@g.com, password: f

The login page can automatically detect whether the user is an admin or a normal user and redirect them to the corresponding page.

4.2.2 Sign Up Page

If the user is a first time user and doesn't have an account, they can click on the sign up button to be redirected to the sign up page,after which they will be able to create an account by entering an email, password and username. Email and password are required to log into the app and the username will be used to display to other user and in the setting page,this is done because though the email is unique, the username is not, along our user to have a professional and recognizable name to refer to each others.

If users click on the sign up page by mistake, there is an "already have an account?" button where they can be redirected back to the login page.

An admin account can not be created from the sign up page. But a normal user can be elevate to admin status by an existing admin using the edit button that they have in their admin page.

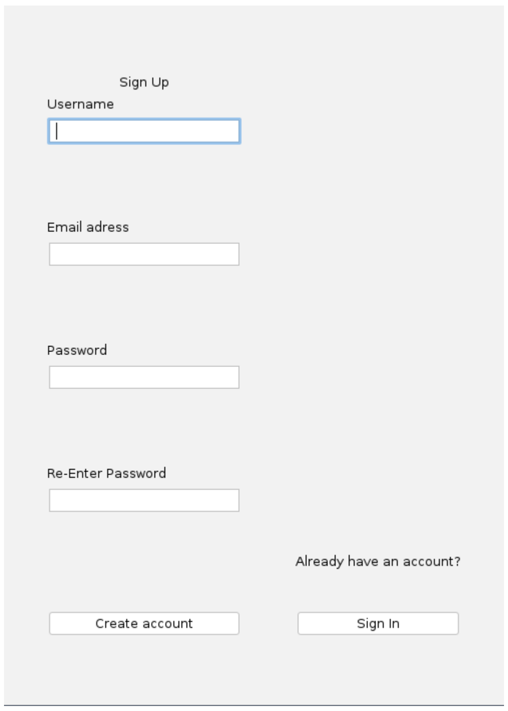

Figure 4.2: Admin page image

4.3 Event Page

Main page of our time scheduler. Here the user has access to most functions of the software.

4.3.1 Calendar View

A live calendar is displayed using the current time of the user's computer as the initial month. Each cell represents a day of the month and can be clicked to open a new frame, showing all events of that day. The cells are highlighted by colours if an event is in our database on that day. The colour changes based on priority from green to red. By clicking the « or » buttons at the top, the user can change the displayed month of the calendar

4.3.2 Side Bar

Includes a button labeled as "New Event" to access the Add Event page where you can create a new event. The Upcoming Events panel shows you a list of your current events. Lastly the Sync button allows the user to synchronize with the remote database.

4.3.3 Adding Events

A frame allowing to the user to add new events to his scheduler. It can be opened either by using the add event button on the main page or by clicking the add button on the show events page. The user input is checked to avoid invalid inputs.

4.3.4 Show Events

A frame showing all events of single day. It can be opened by clicking on a cell in our calendar. When an event from the list is clicked, a new frame called Edit Event with additional information about that event is created.

4.3.5 Edit Events

A frame allowing the user to edit information about an event or delete it. It is almost identical in design to the Add Event frame.

4.4 Database

Our application currently operates on two SQL database.

4.4.1 Remote

Database server hosted by Frankfurt UAS faculty 2 for the course Database taught by Prof. Dr. Christian Rich. Connect to the database using Triet Huynh account assigned to him in the previously mentioned course as learning resource. The database is using Oracle SQL™and is run constantly 24/7. This is the database that the user interacting with when login, sign up or use any of the functionalities in the admin page.

4.4.2 Local

The database is run locally on the user's computer using SQLite version 3036000. SQLite is a light weight implementation of a DBMS, allowing us to store and retrieve data that doesn't require the user to install anything to be able to interact with the local database. Connection to the local database has almost no latency and is reliable. Connection to the local database is only established after the user has finished login or sign up, connection is then terminated when user closed the app.

4.5 Security

4.5.1 Password Encryption

To protect the user from the aftermath of a security breach of our remote server.The user's password is hashed using sha2-256 locally on the user computer via the application. The remote database only stores the hashed password.

4.6 Admin Page

The administrator (admin) has extended rights, an individual GUI with a full view/access to the user profiles in the database and executes commands like changing a username or email and the possibility to completely delete a user from the database. Admins do not have access to event pages. The admin sole purpose is user database administration.

4.6.1 Edit

The admin has the option to edit the username and/or email of a user if it's i.e. not appropriate and the admin feels that it has to be changed or the user wishes for a name change. The changes will be saved in the database and updated on every other part of the app where the email/username has to be displayed.

4.6.2 Delete

The admin also has the capability to delete a user in case he wants his account deleted for personal reasons or if he has an impression of the user violating the terms of service (i.e. misbehavior, profanity). In case this happens, the user will then be fully deleted and non-existent in the database or application.

4.7 Email Function

The emailUtils class contains all the necessary tools to establish a connection to the gmail servers and a method that is used i.e. to send out predefined automated reminders (depending on the reminder selected) or notifications when events are changed/deleted via user emails passed in as parameters to the participants. It has the option to modify the header/subject and the body (main message) of the email (passed in as parameters as well).

4.8 Menu Bar

The Menu Bar has several drop down menus like Export, About, and most importantly the Menu. There we have the access for the user to the Profile Page, Settings, as well as a Log Out and Exit options.

4.8.1 Profile

The Profile page welcomes the user and displays the superficial details as in username and email but also gives the ID number for each user. Additionally the Profile page shows a Contacts section where the user may enter the Contacts page.

4.8.2 Settings

If the User want change any part of their details, they can make those changes themselves by using the Settings page which offers three methods to change each attribute of the current user.

4.8.3 Contacts

The User has a contact list. This page allows you the options to Add and Delete contacts from your personal contact list. They will then appear in the add "Participant" tab in the event creation for a day.

4.8.4 Database

Include two buttons to export and import the local database.

Chapter 5

Diagrams

5.1 Class diagram

Author: Jorge Vanegas A. - 1333459
Co-Author: Minh Triet Huynh - 1370690

The central role is conducted by the User class as it contains the current user information. This is stored in the remote database "DbConnection".

To prompt and enter the software the LoginPage will first check the credentials of User from the database, and if they are not registered the SignUpPage then plays the key role in creating a new users and updating the remote database with the new user's data. To be able to gain access to the admin page a check of the user's credentials similar to one done in Loginpage is also done.

The only difference is in our local database, implemented to reduce queries latency, and its relationship with the CalendarView, as in our main event. For example there the events are locally stored for ease of access and updates.

Chapter 6

Challenges and Solutions

During the development of our project, we ran into different challenges as a team and also faced them individually. In the following section, each member describes their biggest problem(s) and also the solution to it.

6.1 Application availability

Author Huynh Minh Triet - Matriculation number: 1370690

6.1.1 Challenge

When first created our app, we did it locally using mySQL hosted locally on Ante's computer. Resulting in the scenario where functions that deal with the remote database such as: login, sign up or adding events can only be use as long as Ante's personal computer is running. This is a problem for other team members besides from Ante that want to test with how these database related functionalities work, they are dependent on Ante to get their work done.

6.1.2 Solution

Since one of our team member: Triet Huynh is currently taking the Database course at the same time at the university which supply him a remote database hosted by the university, we decided to use that for our application as a remote sever storing the user's data.

The remote database uses the function sys_guid() provided by the Oracle database to assign a globally unique ID for each of the user.This means that no two IDs generated using this method will have the same ID. this function is called whenever a user has successfully signed up to automatically assign them a unique ID.

6.2 Responsiveness and connectivity robustness

Author Huynh Minh Triet - Matriculation number: 1370690

6.2.1 Challenge

At first every interaction with the application that requires stored information is all done through the remote database. This cause delay especially when the throughput of the internet connection is slow. Additionally, the application will come to an abrupt halt if the internet connection is suddenly lost.

Furthermore, the database administrator for the Oracle SQL database that administer the remote server has place a limit of one connection to the database for one user account. This problem is discovered when two members of our team tried to use the application at the same time, wherein they both clicked on the add event button simultaneously causing the application to freeze, stopping it from working.

6.2.2 Solution

Though the case mentioned above is relatively rare but we realized that the more each interaction with the application requires a corresponding query to the remote database the chances of this happening is more often. To solve this we setup a separate local database using *SQLite* that stores all of the user's data which includes events and the contact list.

There are three relations that exist on the local database: the event relation, time relation, and participants relation.

Additionally a sync button is also added allowing the local database to be uploaded and save in the remote database.

6.3 Repetitive typing of the participants' emails

Author: Huynh Minh Triet - Matriculation number: 1370690: Backend
Co-author Jorge Vanegas A.- Matriculation number: 1333459: GUI

6.3.1 Challenge

Earlier version of our application require that each of the participants' emails has to be manually type in for each events that are being created. While is is feasible for small events with one or two participants; however, when the number of participants grow to ten or twenty people the chances of typo when typing the emails are inevitable. This leads to the reminder not being delivered to the participant and made the process of adding event a manually intensive and tedious one.

6.3.2 Solution

By adding an additional page to the user's profile page inside setting called *Contacts.* Participants can be added into or delete from the contact list store in the local database. This contact list is then later shown in the add event page as a scroll panel where the user can choose one or multiple participants to add into that event. This allow for frequent participant to be included with relative ease.

6.4 Offline mode

Author Huynh Minh Triet - Matriculation number: 1370690

6.4.1 Challenge

Due to the application not only require the remote database to authenticate the user in login and sign up but to also download and upload the local database, the app couldn't be use at all when there is no internet connection.

6.4.2 Solution

The latest version of the application now has a offline mode which allow the user to use the app without any internet connection. When the offline mode is selected and if the application has been used before, meaning this is not the first time that the user run the application on the current computer, the application will utilize the current database, allowing them to continue work where they left off. In the case where this is the first time the application is being run on the machine a new blank local database will be create.

In addition with the offline button, a new menu bar button called *database*, which when clicked on will drop down to reveal two more buttons import and export database; which can be use as backup or to transfer the work onto a different machine, e.g., you can export the local database to a USB stick and import it later to a different machine.

In the case where internet connection is restored when the user is in offline mode, they can click on the sync button which will show the *re-authentication page* where it will ask the user to enter their email and password to find the account on the remote database. After which they can continue to work in online mode as usual. The sync button will simply do nothing if internet connection is lost again and the user has to use the import/export database button to transfer their work.

6.5 Calendar View

Author Tim Görß - Matriculation number: 1252200

6.5.1 Challenge

From the very start of our project we decided against a simple list to display current events of our users and instead wanted to do a more interact ability and visually pleasing implementation, similar in design to example Google Calendar.

6.5.2 Solution

We use a JTable to display our calendar, where the columns represent the week-days. A Gregorian Calendar object is used to get the current time of the user and also allow the changing of time. The calendar object also handles leap year problems. Because each month has a different amount of days and the starting weekdays changes as well, we need more cells than actual days and create the calendar for each month dynamically.From the first day on wards, the following cells are filled with a number according to the days of the month. We use a custom DefaultTableCellRenderer to change the colour of the cells according to the priority of the events linked to that day. Each cells is also clickable and opens a new frame showing the events added to that day.

6.6 Menu and Side Bar

Author Jorge Vanegas A. - Matriculation number: 1333459

6.6.1 Challenge

As the many features of the application were being added, we found more and more the need to have a place where they could be accessed. For example the Menu which includes Profile, Settings, Exit and Log Out options, as well as Export, Database, and an About page.

6.6.2 Solution

We concurred on the idea to have a fixed side bar, here we could have our sync functions as well as an add event button. Moreover we could show a list of upcoming events to the user. Most importantly however was the implementation of a Menu Bar where all the above mention features could be accessed.

6.7 Icons

Author Jorge Vanegas A. - Matriculation number: 1333459

6.7.1 Challenge

GUI with logos and icons would look more pleasing to the user. However we found that Java swing makes it on one side easier to work with the GUI forms, but on the other side a bit more complicated to integrate icons into for example the buttons. Personally could never figure out if it was working with Git or if it was Java swing. We concluded to leave the generic buttons as they are.

6.8 Invisible database in JTable

Author Ante Maric - Matriculation number: 1273904

6.8.1 Challenge

While creating the admin GUI that had to display the user database where the admin could edit/delete users from it, the database wouldn't show up in the GUI (empty JTable - invisible database) even though the database was implemented correctly in the code (Tested with System.out.println(); functions).

6.8.2 Solution

After some try and fail options, thoughts about the code being wrong, the solution to it was to manually add the columns and name them above the main section of the actual code to make them visible in the JTable. The database was finally displayed properly and the users could be selected for further function implementation.

6.9 Database not updating changes

Author Ante Maric - Matriculation number: 1273904

6.9.1 Challenge

After the admin GUI was created and the database fully displayed and ready to be worked on and after successfully implementing the delete function, we started working on implementing the edit function where the admin can select a user from the database and change his username and/or email. After coding for a while and testing out the implemented SQL queries ("UPDATE XYZ FROM ABC WHERE 123"), the database should be updating the users according to our changes made from the GUI options but it did not. We didn't receive any errors on the execution of the program and the function seemed to work in theory, but as shown in the database it didn't change anything.

6.9.2 Solution

Days have passed and we tried several different changes/options but we didn't get any good results. After additional changes, we received an SQL exception (unique constraint) where it showed us that some attributes in the database where set up to not allow any duplicates with the changes so we had to change them in order to allow the manual edit from our admin panel. As we fixed the database problems, we had to also make some changes to the SQL Queries and combine them with some get methods from our edit function and everything finally updated on the database when we pressed the apply changes button.

6.10 Emails not being received and accounts getting banned

Author Ante Maric - Matriculation number: 1273904

6.10.1 Challenge

While testing out the email API, we created some dummy accounts on gmail and tried to send out test emails to see if they were properly sent out and received by our account. The emails were sent out from our program, but there were no emails in the mailbox. Email accounts couldn't be used anymore for testing as they were getting banned.

6.10.2 Solution

After some research, we found out that you have to turn ON the "Allow less secure apps to access your account" in the gmail settings to receive the emails on our gmail account. Afterwards, everything worked as it should. After fixing the problem with the receiving emails, the accounts were getting target banned after a few days as they received some test emails and they have been marked as unsafe from gmail as we don't have any licences as other apps/programs have to prove that we are not malicious (problem can't be solved).

Chapter 7

Conclusion

Author Ante Maric - Matriculation number: 1273904

With this project, we had our ups and downs. We learned a lot about team work and trusting each other with the tasks we have been given. If someone struggled and was stuck on a task, there was always someone from the group to speak to/ask for help. This way we grew as a team and had a lot of fun as well working on our project.

Each and everyone in our group was working hard and faced their own challenges on their way but we all persisted and we are proud of what we have accomplished in this short amount of time.

We learned a lot of practical things not only in terms of Java and it's functionalities, but also expanded our knowledge on databases, creating GUIs, email API, network connection, understanding the Calendar and time and what can be done with it et cetera.

Everything we have learned will be of great usage in our future endeavors and we have learned that you have to be adaptive and inventive facing difficulties and changes.

Never give up and stay persistent, you will eventually overcome your problems and struggles.

Chapter 8

Sources

8.1 *Ante Maric - Matriculation number: 1273904*

8.1.1 Email API

[1]link: https://www.javatpoint.com/example-of-sending-email-using-java-mail-api
[2]link: https://www.youtube.com/watch?v=A7HAB5whD6I

[1]Author: © Copyright 2011-2021 www.javatpoint.com. All rights reserved. Developed by JavaTpoint.
[2]Author: Genuine Coder
Last accessed: on the 11th of February, 2022 - 8.11 PM

8.1.2 Reminder functionality

[2] Youtube tutorial about timers and timerTasks well exlplained

[1]*link:* https://www.baeldung.com/java-timer-and-timertask
Author: Eugen Paraschiv - Baeldung

[2]*link:* https://youtu.be/QEF62Fm81h4
Author: Bro Code

Last accessed: on the 11th of February, 2022 - 8.26 PM

8.2 *Huynh Minh Triet - Matriculation number: 1370690*

8.2.1 Jfilechooser file type filter

link: https://www.youtube.com/watch?v=lFkYt2jKrYc

8.3 *Jorge Vanegas A. - Matriculation number: 1333459*

8.3.1 Menu Bar

https://www.youtube.com/watch?v=dwLkDGm5EBc
Last accessed: on the 26th of January, 2022

List of Figures