Project

Cloud eBook Converter

Aaron Siessl

Tutor: Michael Wild

Class 5 I a

TFO “Max Valier” Bozen

School Year 2013/2014

Contents

1 Introduction

An eBook is a book in electronic format. It can be downloaded to a computer, PC, tablet, smartphone or any other kind of reading device and be read there. Over the years, the eBook market has grown from a relatively small niche to a multibillion-dollar mass market. The total eBook revenues went from 64 million $ in 2008 to about 3 billion $ in 2012. This is an increase by a factor of close to 50 in just a few years.

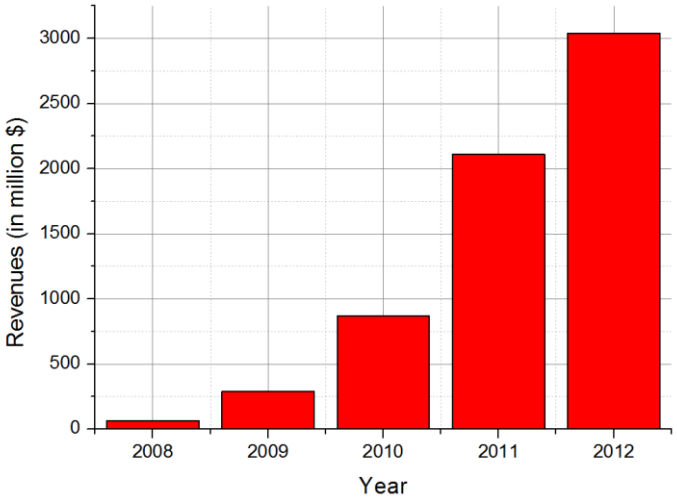

Figure 1: Revenues in milion $ a year

Developers around the world deployed a couple of different file formats for eBooks, but none of them became a standard. Companies like Amazon developed their own formats like MOBI and AZW, meanwhile open source developers created open formats like EPUB. eBook readers today ideally should support about ten different formats, but unfortunately, not all of the eReaders support them.

1.1 The Application

To abstract these problems I created a web application, called “Graphitebook”, which brings a solution for all eBook fans. Graphitebook is a free web based application developed for managing your own eBooks on a cloud storage. With this web application, you can organize your entire eBook library any way that you choose, and synchronize all your books with your favourite devices.

You can upload your eBooks in any file-format you want. When you send the book from the application to your eReader, the application checks which file formats your eReader supports and converts the eBook to a suitable format, before sending it to your device. You do not have to check if you bought a book in the right format anymore. You can easily register all your devices in the application and then synchronise all your eBooks with a single click.

To receive all these features, you need to subscribe to the application by creating an account, so all your eBooks are private and no one can access your personal book library. In addition to this web application, I implemented a java side administrative application, which manages the incoming data.

1.2 Software specifications

The development of the web-application requires many different programming languages, libraries and a lot of different software. In the following list I write the most important libraries and languages that I used to develop the web-application and the administrative tool, to manage the data:

Programming languages and libraries:	
HTML Version 5	
CSS Version 3	
PHP 5.5	The server-side software logic is written in PHP 5.5. This new version of PHP has a complete reworked hashing API, which I use. PHP is known for its strong Mail-API and integration of Postfix. With PHP it is also easier to read or write data byte-by-byte than Java. This is precisely the reason why PHP is the most appropriate programming language for this project.
JQuery 1.9	JQuery simplifies JavaScript programming, which animates the application and makes it more user-friendly.
jQuery UI	jQuery UI is a jQuery library, that's provides a powerful set of user interface elements, widgets and user interface interactions.
Tooltipster	Tooltipster is a powerful jQuery plugin, which enables easily to create modern designed tooltips.
jQuery AJAX Module	Parts of the web application use AJAX to communicate with the web-server without reloading the homepage.
Dropzone	Dropzone is a jQuery plugin, which handles all file uploads. Dropzone uploads files parallel over AJAX.
jResize	jResize is a responsive web development tool, built in jQuery to create websites optimized for PC, tablet and smartphone.
Java SE 8	The administrative part of the web-application is written in Java with a client-server structure. The Java-Server is responsible for backups, data maintenance and data management.
SQL2O[1]	SQL2O is a free java framework, that helps developers to execute SQL-Statements using JDBC. I used this library as simple ORM for the administration application.

[1] Development Page: www.sql2o.org

Apache POI[2]	Apache POI is a strong Java written library that provides the export of data in Microsoft-Office formats. The administration application uses this library to export data into Microsoft Excel 2010 (*.xlsx) format.
SLF4J[3]	SLF4J is a Java written logging facade. This library uses the administration application, to log all interactions of administrators into log-files. As SLF4J engine, I use Logback[4].

The application needs **Apache 2.2** and **MySQL 5.6**. The server- and the clientside administration program needs the **Java Runtime Environment 8**.
During the development phase, I used **Eclipse 4.2** (Kepler) as development tool and **MySQL Workbench 6.0** as Database Query program and Entity-Relationship-Modell editor.

1.3 Schedules

Summer 2013: Development of the eBook converter between the MOBI and EPUB format.
September / November 2013: Implementation of registration system and web design. Planning eBook parser and conversion.
December 2013: Improvement of eBook converter and implementation of new conversion methods for other file formats.
By the beginning of January 2014: Upload of eBooks and implementation of Cloud-Storage.
By the end of January 2014: Implementation of metadata editor and conversion between file formats.
By the end of February 2014: Implementation of synchronization System.
March 2014: Tweaking and improvement.

1.4 Security aspects

Web applications present a complex set of security issues for developers. The most secure and hack-resilient applications are those that have been built from scratch with security in mind. This is the reason why Graphitebook attach importance to many security issues.

[2] Development Page: poi.apache.org
[3] Development Page: www.slf4j.org
[4] Development Page: logback.qos.ch

1.4.1 Private data encryption

In a world where the secure storage of data becomes more and more important, it is necessary to prevent hackers from getting private data of users. This is the reason why Graphitebook saves all private data of users encoded with bcrypt and not MD5, SHA1, SHA256, SHA513 or SHA-3. These last ones are all hash functions, designed to calculate a digest of huge amounts of data in short time. A modern server can brute force the MD5 hash in a short time with rainbow tables. Bcrypt solves these problems: the most important advantage of bcrypt is that it is very slow. It has salts built-in to prevent rainbow table attacks. PHP integrates in its new version 5.5 this complete new hashing API. A bcrypt hash might look something like this:

$2a$10$3guHuNf.HNSr3X1B3jBQPOhba5YCyFpnueRpJ79/IgqC1VdmCM0XS

- "2a" identifies the bcrypt algorithm version used
- "10" is the cost factor: 2^{10} iterations are used
- The rest of the hash are the salt (first 22 characters decoded to a 16-byte value) and the cipher text.

1.4.2 File Uploads

Graphitebook uses "HTTP file uploads" to upload eBooks to the cloud storage. This is a big security issue. The application needs to prevent security attacks like self written PHP scripts or ZIP bombs. For this reason, the upload folder is a folder, where no file can be executed. To prevent ZIP bomb attacks, Graphitebook parses the comprimized EPUB format, before it extracts data from the archive. Graphitebook uses four security steps to check, if an uploaded file is malware or not:

1. Block files with dangerous or unknown extensions
2. Validate the mimetype and check the maximum uploaded file size
3. Block the upload folder with .htaccess and Apache EventHandler

To prevent simple ZIP bombs, PHP offers a secure ZIP-protection-library.

1.4.3 SQL Injection

SQL Injection is the most used web attack mechanisms to steal data. It is a technique where an attacker creates or alters existing SQL commands to get hidden data, to override them or even more dangerous: to execute system level commands on the database host. To prevent

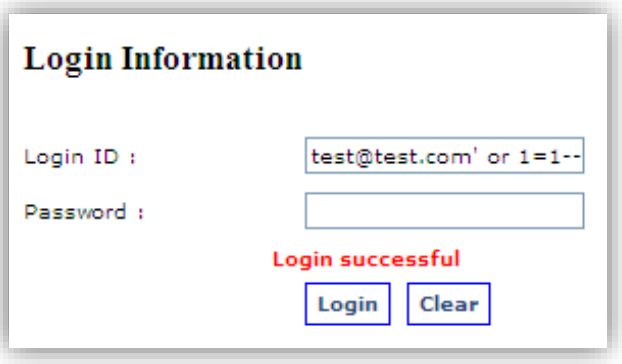

Figure 2: A simple example of SQL injection.

these attacks, Graphitebook uses **prepared statements**. These statements were written to the database, before the user-input gets included as parameter. All parameters are bound with the bind function of mysqli, but before, they get checked with regular expression, if the user input accepts the rules.
As additional defence method, the application has more database users, with strict permissions. Therefore, in case of an attack, a hacker is able to manipulate some tables of the database, but not all.

2 Login System & Webdesign

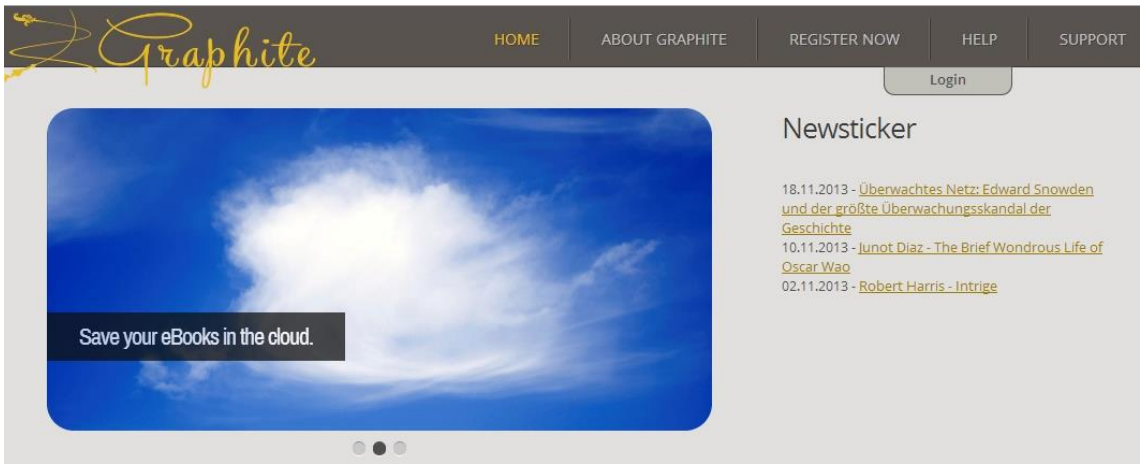

Figure 3: Screenshot of the main screen

This screen appears, when a user enters the web application the first time. A little description text with some promotion slides, pictures and videos show the features of Graphitebook. From here, the visitor can navigate to different services. He can subscribe to the service, login, but also create support tickets, if he has a problem or wants to get in contact with the web administrator.

On the home page Graphitebook continuously posts new articles about the web application, but also news about new books. These articles should help the users to find new interesting eBooks and new authors to promote their products. So Graphitebook offers a free platform for authors and readers to exchange their ideas.

2.1 Modern User Interface

In the era of the digital revolution web portals are becoming always more and more important. It is necessary to develop an efficient web design that provides information in precise and illustrated manner. Good designs should always look to use good colour combinations and ensure that the design is easy to navigate through and things are easy to find on the page, because it helps the user to save time. **Usability** and **simplicity** are the golden rules in web design. For this reason, I always kept the golden rules in mind, while I was developing the application.

To test if the application is written in an intuitive way, I tested my milestones with test takers that did not have any explanation of the functions. Only test takers can help the programmer, to check if functions are easy to find or not.

2.2 Account Registration and Login-System

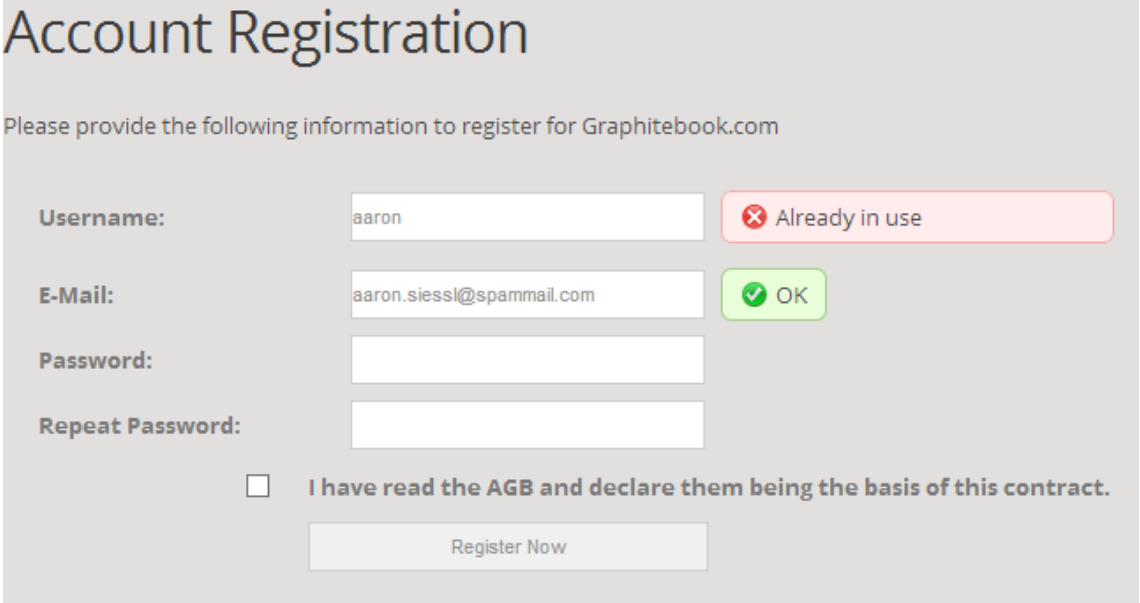

Figure 4: The account registration form

When a user wants to create a new account, they need to fill-in the registration form. Every user has their own account name, email address and password. When a user submits the registration form, they gets an activation email to the specified mail address, where he has to click a link to activate the account.

When a user writes data in the input fields, the applications sends an AJAX request to the server to validate the data. So a user gets informed while filling the form, if the username is already in use or the password does not follow the rules.

When a user submits the form, the server checks, for security reasons, a second time, if everything is correct.

When a user logs in, he has the possibility to log in automatically from now on. This function creates two Cookies in his browser that saves the login information. The first Cookie saves the Account-ID of an existing user encrypted with Base64. The second Cookie saves a part of the account name, a part of the user email address and a part of the encrypted user password. These informations are hashed and then encrypted using bcrypt.

When a user activates the “Auto-Login” function he is automatically logged in for the next 30 days (set in configuration file), but only if he does not modify his account name, email or password. Also when a user logs manually out, the auto-login will be disabled due to security reasons.

2.3 Web statistics

Every single request on the web application will be logged anonymously in the database to create daily new statistics of the usage. The application logs which pages were requested frequently, but also on which days how many users opened the homepage. The application logs also from which websites (for example facebook.com or google.com) the user is coming from and which browser language he uses.

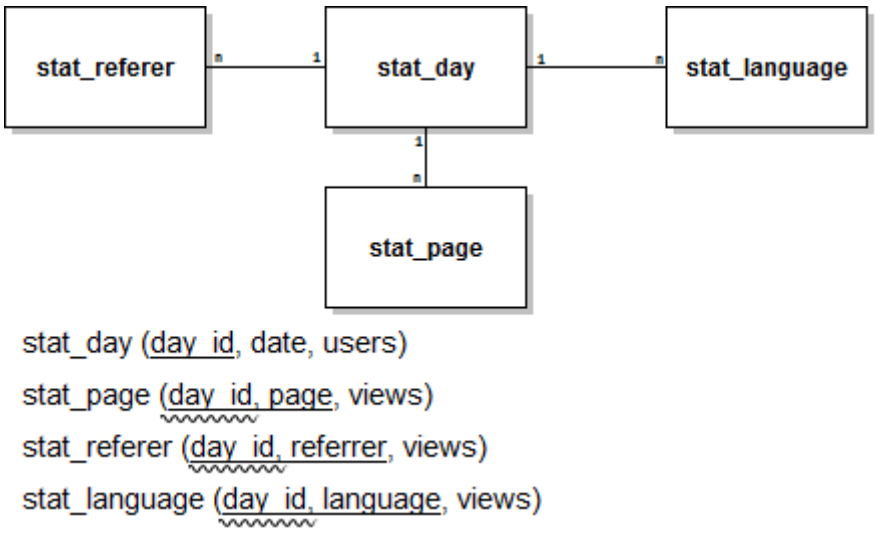

Figure 5: Database schema of the web statistic system

The data is only visible in the administration program, to make sure, that no normal user can access this sensitive data. For legal reasons the statistics are completely anonymous. It isn't possible to find out, the IP address or other private data of users. Graphitebook uses the web statistics to improve the web-portal and to verify from which countries the users send requests and which new languages Graphitebook should add.

3 The Cloud Service

3.1 The libraries

Figure 6: This screen appears, after logging in

After logging in, the user sees his eBooks in a grid. On the left side of the screen, is a list of his libraries. Here a user can categorize his eBooks. These libraries can be added, modified and deleted. When a user deletes a library, all eBooks in the library are also deleted. By clicking on a library, the book-grid reloads and shows all books of this library. With the mouse a user can drag-and-drop the libraries and sort them.

All communication with the server is over AJAX. This means, that the browser never reloads the whole page, but sends single data-modules. This feature is a big benefit for the usability of the page.

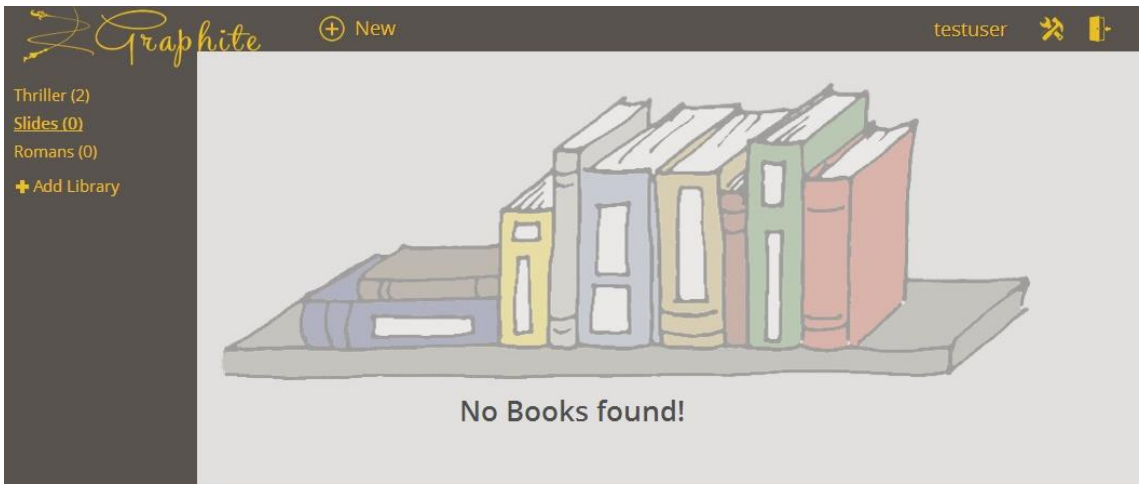

Figure 7: This screen appears, if someone has an empty library

3.2 The eBook-Grid

In the book grid, every eBook is represented by its cover. When a user moves his mouse over a book, he can choose various operations:

- **Edit**: Choosing this option, a dialog appears, where the user can modify the cover, author, publisher or title of the eBook. When a user modifies their metadata, it will be saved immediately in the book-file.
- **Delete**: With this function, a user can delete the selected book from his personal library. When a user deletes a book, the book will be deleted completely from the system, so it is not possible to restore it. As a book requires a few hundreds of kilobyte or megabyte, it would cost too much memory to save all deleted books.
- **Synchronize**: Choosing this option, a dialog appears, where the user can synchronize the selected eBook to his devices. He can choose if he wants to send the book to the device immediately or download it in a desired format.

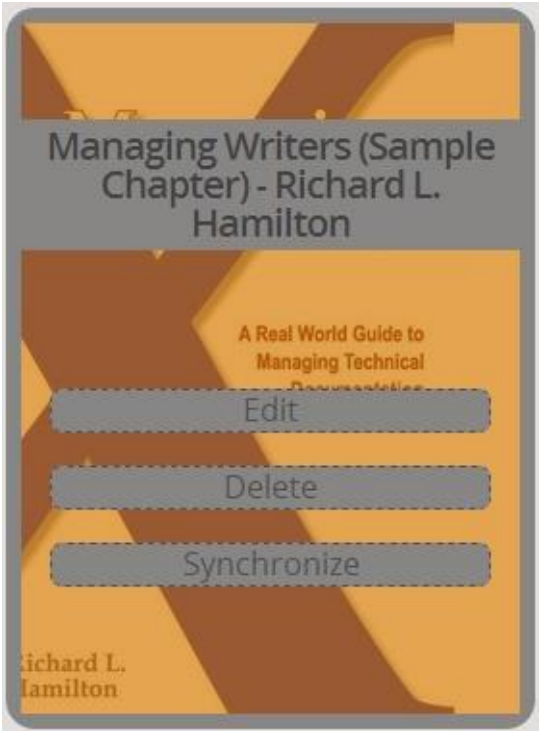

Figure 8: This screen appears, when a user moves the mouse over an book in

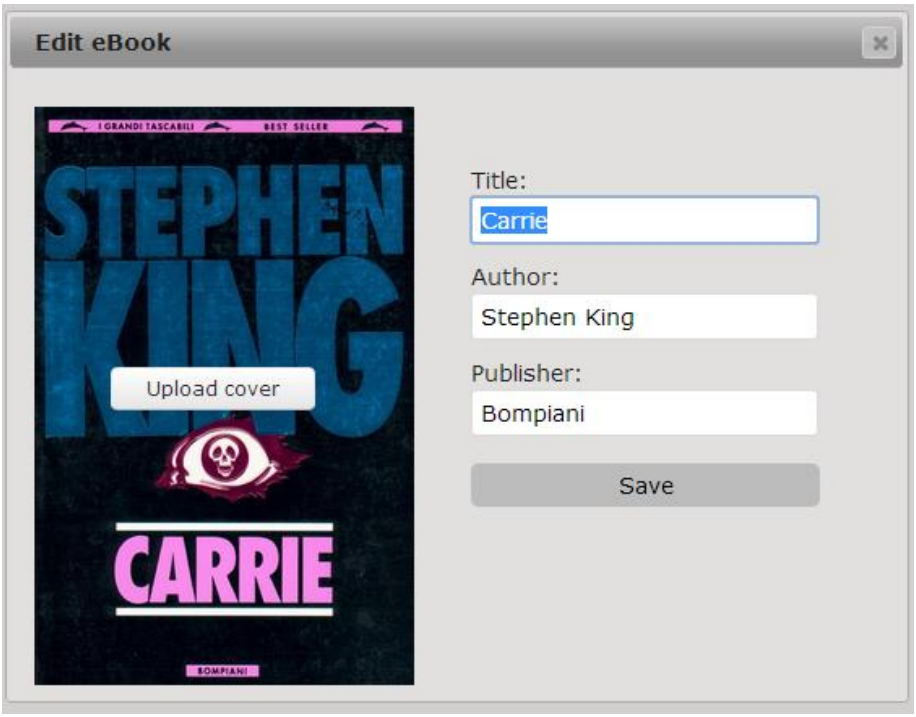

Figure 9: The edit dialog

3.3 The synchronization dialog

At the synchronization dialog, the user can download the selected eBook or send it to his devices. At the left side of the dialog is the list of all registered devices, which can be added, modified or deleted in the account settings.

When a user chooses the Download item from the left list, at the right side of the dialog a 2-column list with all possible download formats opens. When a user moves his mouse over a format, Graphitebook shows some hints about the selected format: for example, for which platforms and devices this format is common or which programs can open the format.

When a user clicks on a format, Graphitebook fetches the selected eBook, converts it into the desired format and downloads it, without reloading the page.

As download, Graphitebook supports only the six most common formats: MOBI, HTMLZ, EPUB, TXT, PDF and AZW Version 3. It would be too confusing, when all supported formats are listed there. For this reason, Graphitebook supports all formats in import, but only six at export. Almost all eReader that support AZW1 or AZW2 support also AZW3.

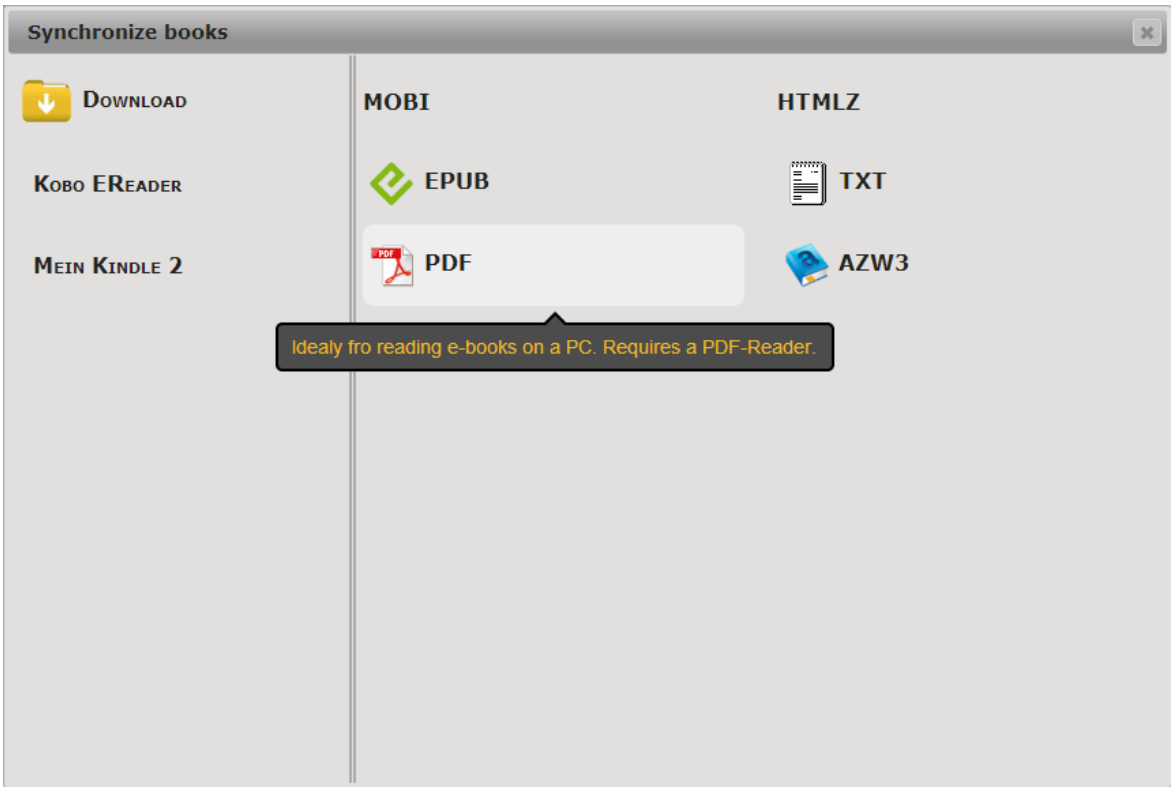

Figure 10: The user receives proposals, which format is suitable for his device.

4 eBook synchronization

The eBook synchronization is based on the email synchronization protocol of many eBook readers. Every user of an eReader has his own mail address. For example, Amazon Kindle Clients have as e-mail address [name].[surname]@kindle.com. When an authorized mail user sends an email to this mail address with attachment, the attached files are downloaded to the linked device. This principle uses Graphitebook. It sends a mail from synchronize@graphitebook.com to the corresponding eReader address, with the book in an attachment. When the eReader connects to the internet the next time, he asks the provider (for example Amazon Cloud Service) if he has new mails and downloads the eBooks.

To prevent misbehaviour of users and sending spam mails, Graphitebook has some security aspects:

- Graphitebook accepts only a few domains. This means, that a user cannot insert as email address a “@gmail.com” address, because Gmail is not an authorized eReader-producer.
- When someone sends a mail to a non existing user, Graphitebook gets an automatic generated response message, and the user needs to edit the address, before synchronizing the next time.
- In the case of fraudulent use of such an authorisation, or if there is suspicion of such a case, Graphitebook is entitled to close the account.

Unfortunately, this method of synchronization is only possible for devices that support the send-over-mail function. For all other devices, mainly cheap eReaders, there does not exist any online gateway to the device.

When a user has this type of eReader, he must download the book in the supported format and copy it manually to the disk of the eReader. Some eReaders have also a browser build in, where they can connect to Graphitebook. Also for these users, Graphitebook is a good solution to manage their own books and have them to the cloud.

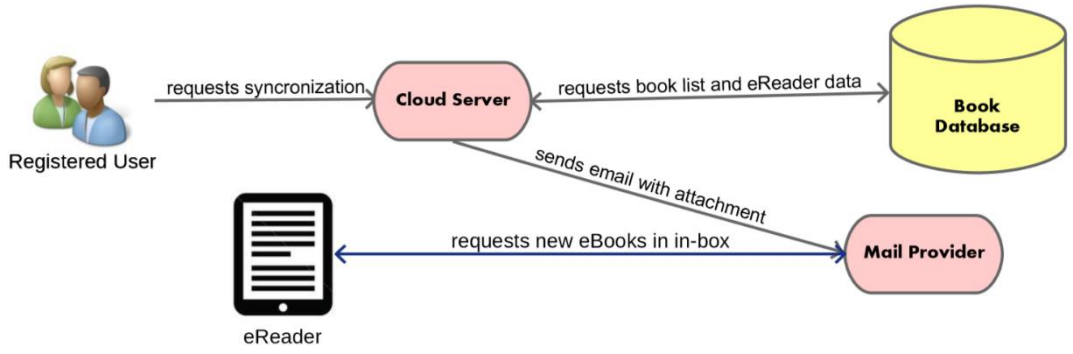

Figure 11: The principle of synchronization

5 eBook conversion

The book conversion is one of the most important features of my application, but also one of the hardest parts for coding. Graphitebook supports various eBook file formats. A user can upload his eBooks in 10 different formats, but they can also have different versions. For example, EPUB has EPUB 1.0, EPUB 2.0 and EPUB 3.0. The application has to know, which version is used. This means, that the converter needs to support about 25 different types. To realize such a big converter, I decided to create a converter in form of a star. This means, that every file is converted into its own file format (called Graphite-eBook) and from there converted in the output format. The improvement of this method is, to create for every file-format only a parser (reader) and a writer module. If I want to add a new format, I only need to write these two modules, to convert it into all other formats. A disadvantage of this principle is that it is slower than converting directly from one format to another.

The converter needs about 2 – 10 seconds to convert a book with about 200 pages into another format. The fastest conversion is the conversion from TXT to another format, because in a TXT file the data is simple to read. The slowest conversion method is reading from a PDF, because there the application needs to parse the PDF intern structure, which is quite complicated and needs much performance.

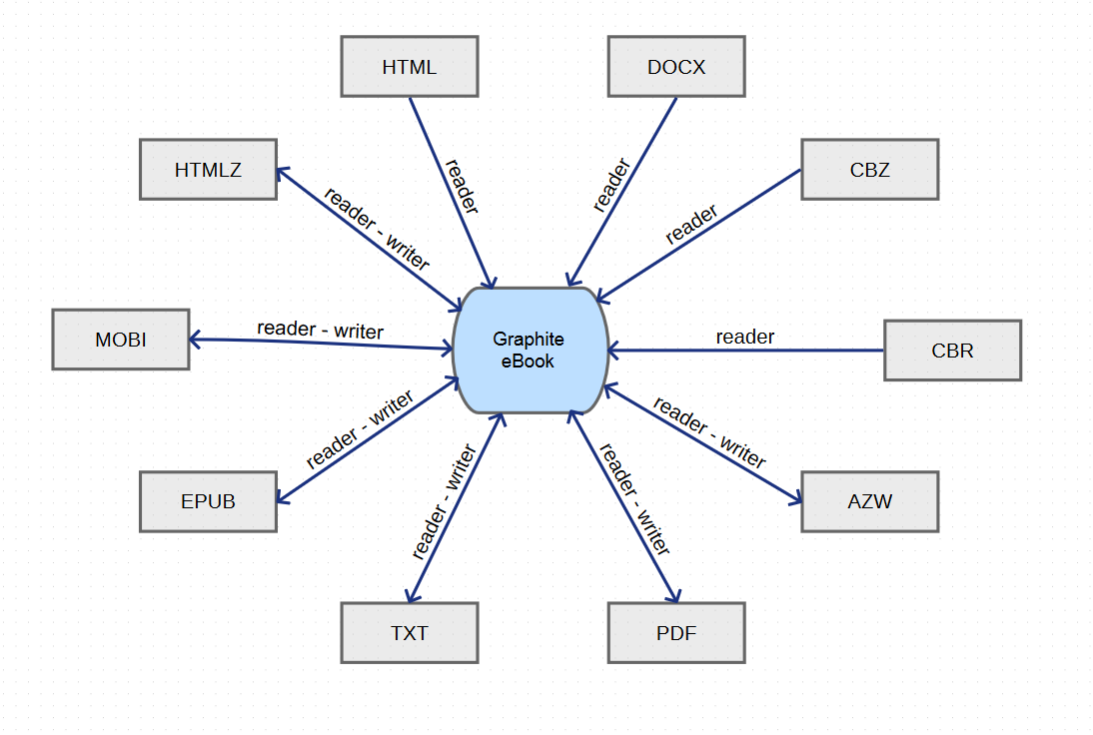

Figure 12: Every file gets converted in the Graphite-format, before converting it to the output format.

5.1 Class Diagram

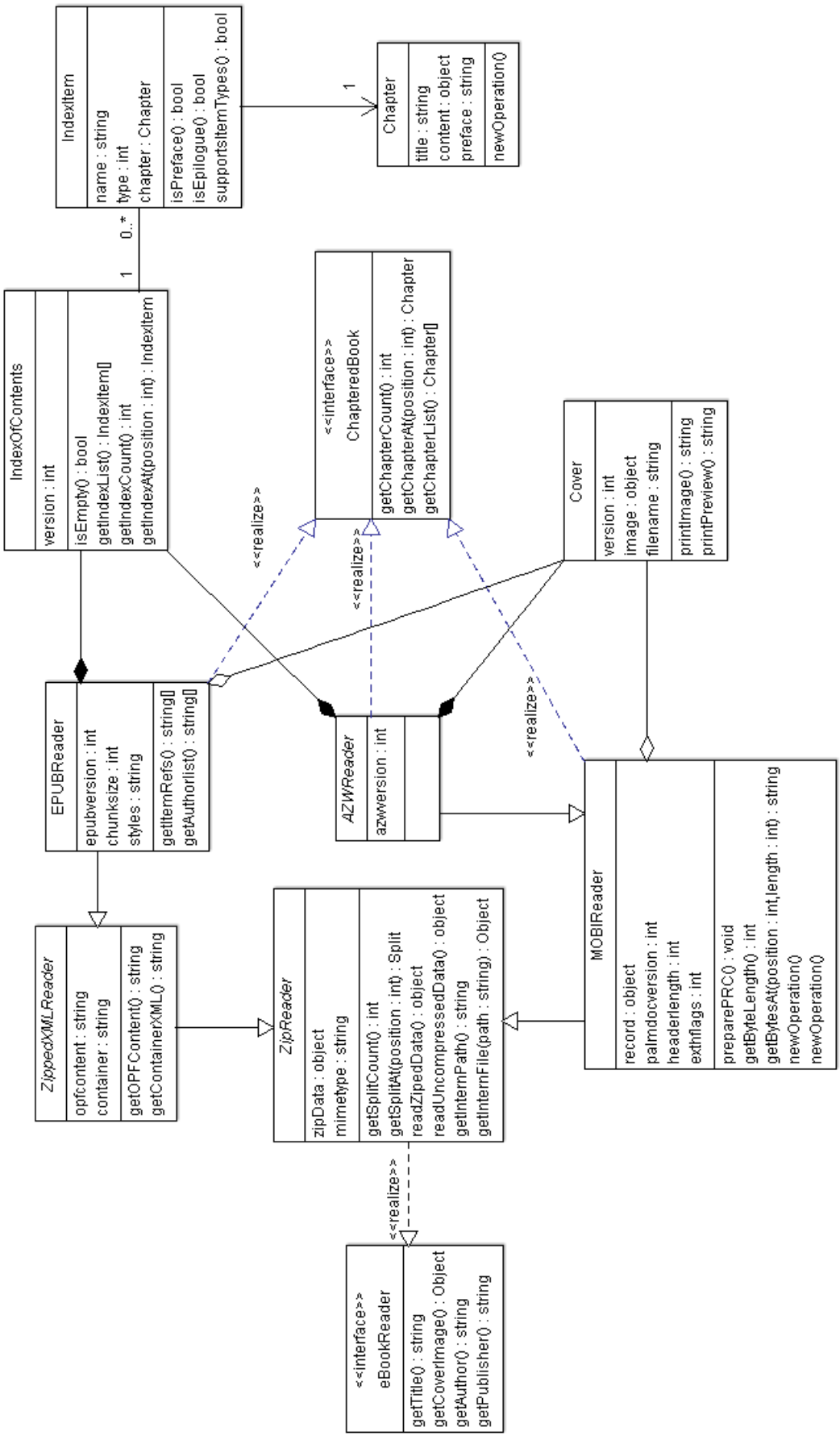

Figure 13: The class diagram is a clipping from the book parser. The diagram contains only 3 formats: AZW, MOBI and EPUB.

6 Database

A good database design is essential for a high performance application, just like an aerodynamic body is important to an airplane. If it isn't aerodynamic, it would slow down, like a bad database design. It`s necessary to save as much data as possible, without saving data several times.

Beyond the issue of performance, a good database should be easy to maintain and to expand. Graphitebook uses the MySQL database server with InnoDB engine. Not only user data, but also news, statistics, maintenance informations and help pages are saved in the database. To prevent security issues, the application supports different database users with diferent privileges. If a database user modifies account data, he can only see and modify the account table, but not other database tables. If a hacker finds a security issue in the application, he can only crash a part of the database and the administrators can easier check, where the issue is.

The following Entity Relationship Diagram (ER) describes the database of Graphitebook. The diagram was created in MySQL Workbench with the own MySQL-Workbench-ER-Annotation, because is more specific, than the classic ER-Diagram in Chen annotation and adjusted for the MySQL Database Software. This diagram specifies not only the tables and their structure, but also the users and their privileges, triggers and stored procedures.

```
CREATE USER graphite_hp IDENTIFIED BY '***************';
GRANT INSERT, SELECT ON TABLE stat_day TO graphite_hp;
GRANT INSERT, SELECT, UPDATE ON TABLE stat_language TO graphite_hp;
GRANT INSERT, SELECT, UPDATE ON TABLE stat_page TO graphite_hp;
GRANT INSERT, SELECT, UPDATE ON TABLE stat_referer TO graphite_hp;
GRANT SELECT ON TABLE strings TO graphite_hp;
GRANT SELECT ON TABLE news TO graphite_hp;
GRANT SELECT ON TABLE languages TO graphite_hp;
```

Figure 14: An example database user. This user is used for the all data at the start page. He can only access general data like language strings and has no access to user data. This user creates also the web statistics.

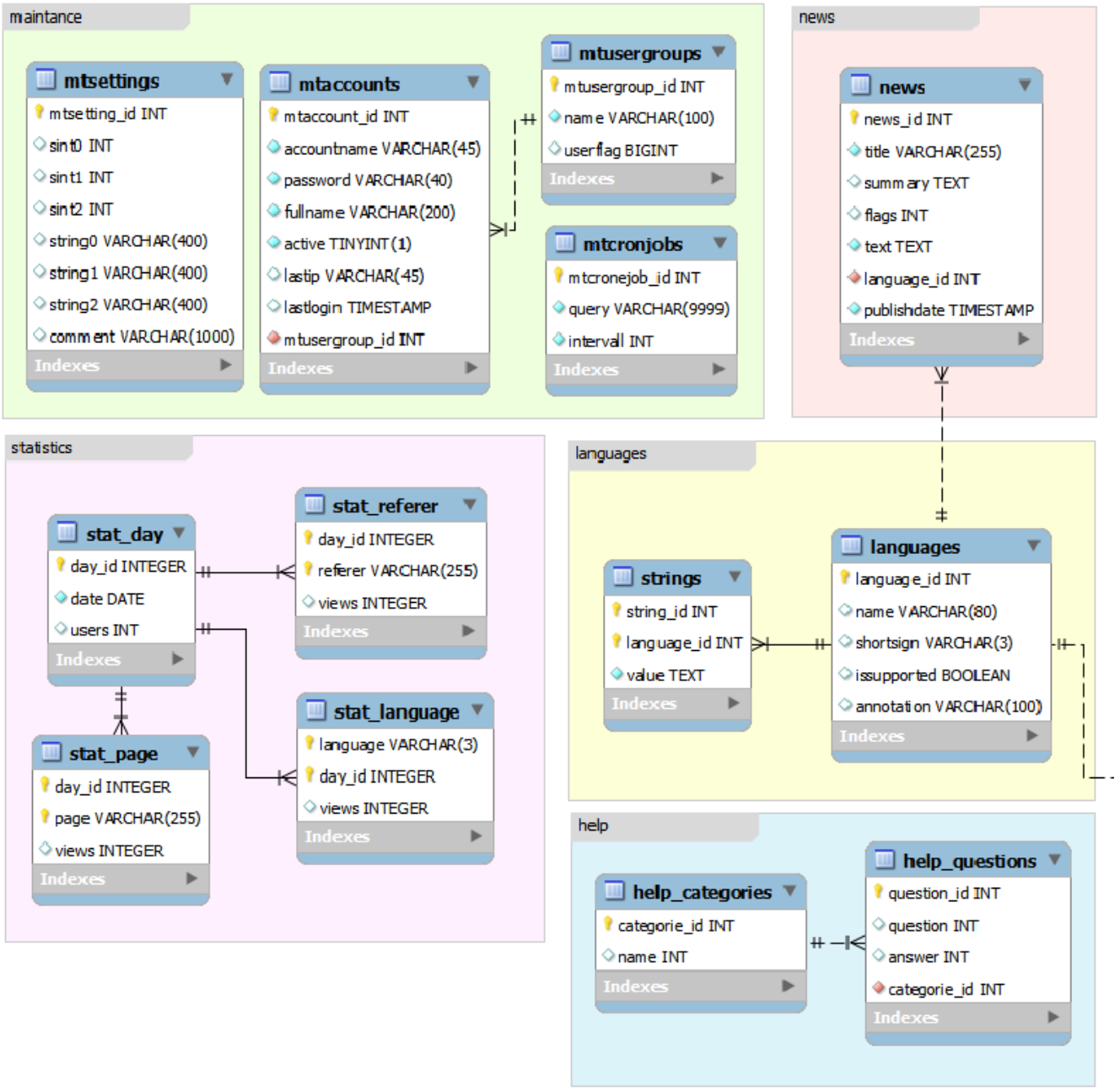

Figure 15: This part of the ER Diagram describes general data that gets saved in the database. Every layer (block in a different color containing the tables) regroups the tables.

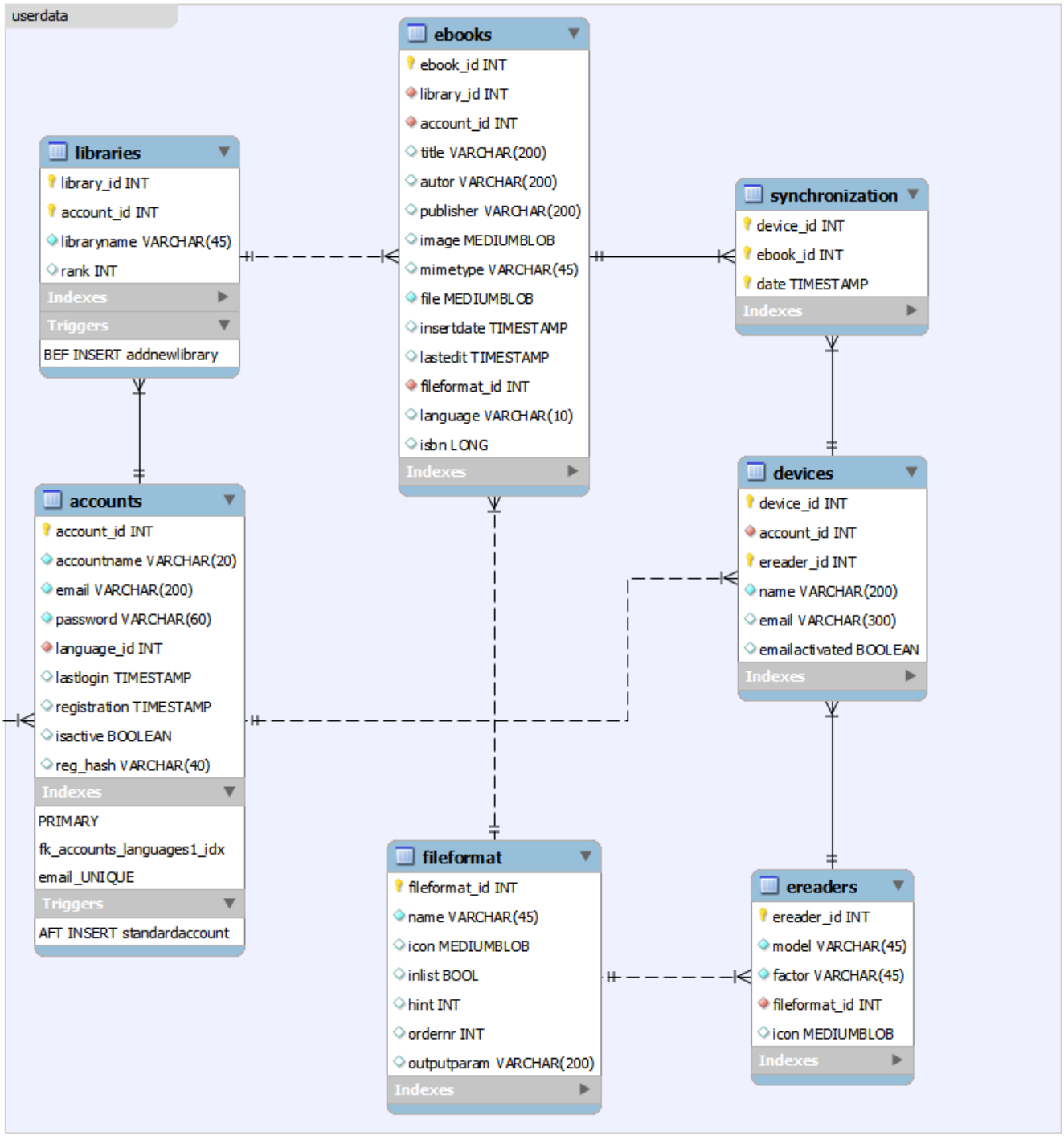

Figure 16: This part of the ER diagram bescribes, how the user data gets saved in the database.

7 Administration tool

The administration tool helps the developer of the platform, to maintain the web portal. This platform is written in Java Version 8 and implements a Server-Client-Model. Every administrator has their own maintenance account (divided from the web portal account) from which he can enter the platform. Every account is associated to a group, which has different permissions. For example, reporters have the permission to create and modify news articles, but they cannot watch the system backups.

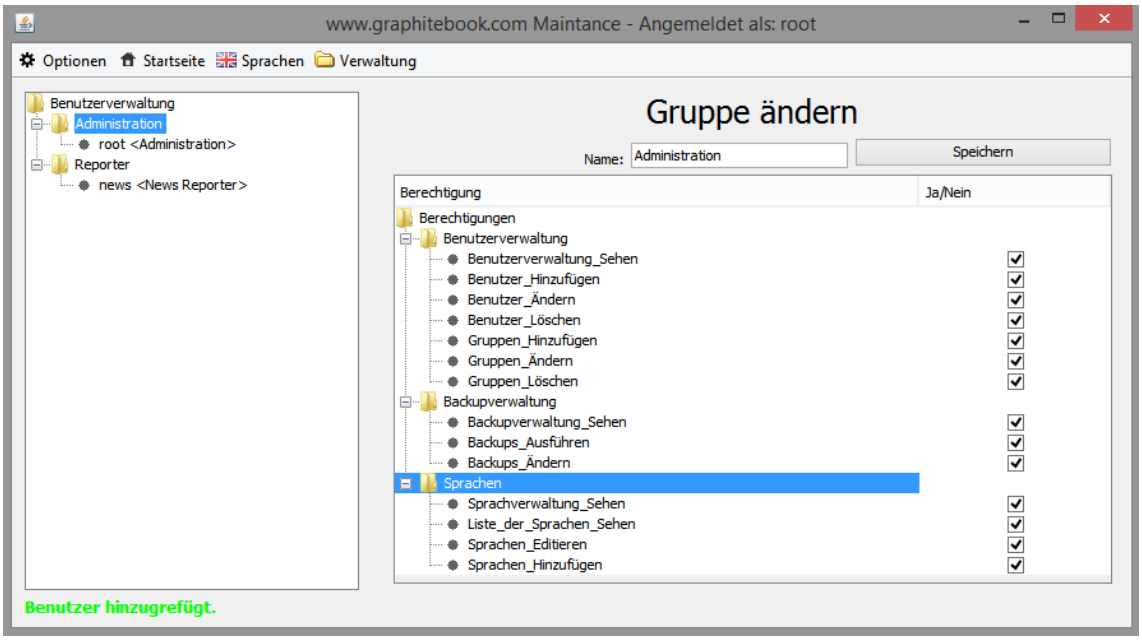

Figure 17: In the administration program i can add maintenance users and give them specific rights.

The administration tool is written in German. There is no translation in other languages planned. It actually supports only a few functions, needed to maintain the application, but for the feature, it will expand it by new functions.

When an administrator logs in, the first thing he see is the statistic page. Here he can see all the captured web statistics, like how many users joined on the web portal the last days or from where they are. From this start page, an administrator can browse through the various functions.

One function is the news section. From there, an administrator can post new news articles to the homepage or modify already existing articles. The articles are categorized in the different languages so it is also possible to write a news article in English without a German translation. In addition, it is also possible to set up the time, when the article gets published. An author can write a Happy New Year Article during the year, while the system publishes it on 31 of December.

7.1 Language modification

An important function of the administration tool is the language service. In this tab, an administrator can modify every string that is printed in the web application in every language. From this screen an administrator can also add new strings of languages and begin to translate the strings into the new language. Only when a language has every string translated, can the language be activated. A very nice feature in this screen is the split screen. In this screen a administrator can combine the strings of two languages. With this function an administrator can check if every information of the string is included also in the translations.

7.2 Backups

A very important function of he administration tool is he backup service. The administrator can set up, when the system gets backuped. I actually configured a daily backup. This service backups the whole database, where the userdata is saved, but doesn't save configuration files or serverfiles on the harddisk, because a complete server backup is made by an external tool every week.

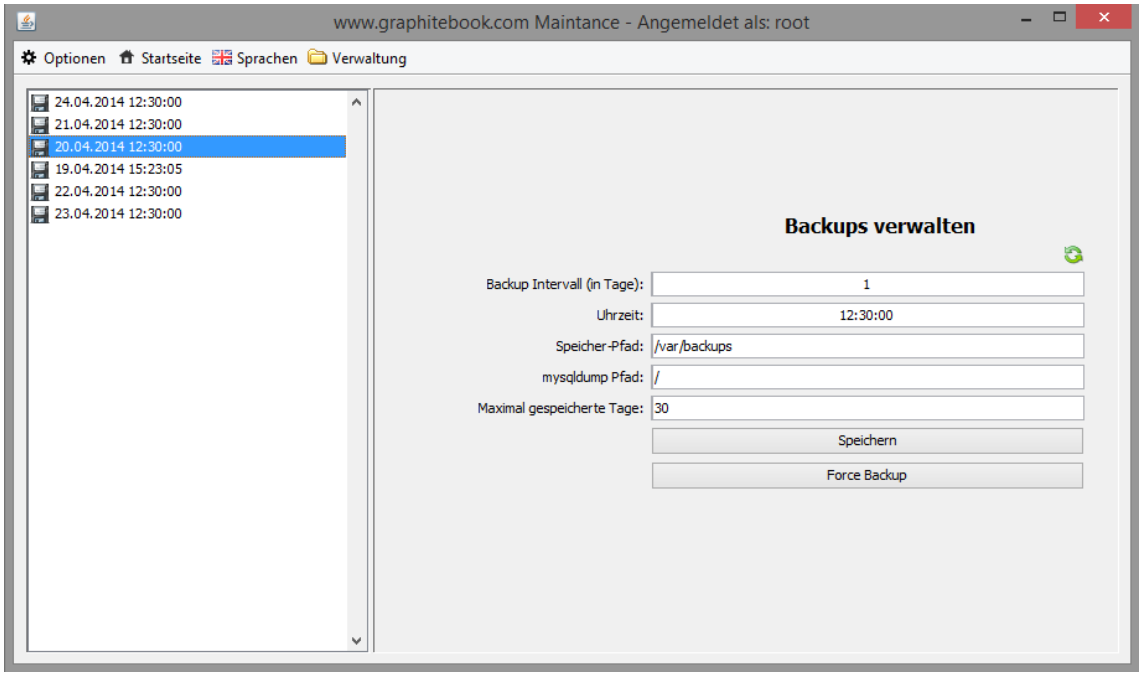

Figure 18: In this screen an administrator gets informed, when wher the last system backups done and on which time the system is getting frequently backuped. In this example the system backups every day at 12:30 to /var/backups/. From there they will be exported to a external hard disk.

8 Expandability

Graphitebook is a very extensive and complex application for a single developer. For this reason, I had to cut some features during development. I would like to mention in this chapter some expansion options that can potentially be implemented in the future. I have invested a lot of time into this project and I am still convinced that this software could simplify the book management for many persons that read eBooks. For this reason, I want the project to live on also after the Matura and therefore I shall publish it on the World Wide Web in the near future.

8.1 Smartphone development

GraphiteBook is a cloud application, which should be accessible from everywhere. For this reason, a smartphone support is urgently needed and planned for the coming summer. All data access will be done over the webserver, while the user interface in the app will use the responsive UI of every operation system.

I've already started to plan the application and chose to use the library "Codename One". With this open source library, I'll write the Java code once and can compile it for Android, IOS, Windows Phone and Blackberry OS. The library translates everything from Java to the responsive system design. The compiler for IOS (which needs to be an Apple IOS system), is in the cloud, so I do not need any IOS operation system to develop, because Codename One gives me already to opportunity to compile IOS applications on they're server.

Figure 19: Codename One is used by many big companies. Over 200.000 developers are using the open source SDK of Codename One.

8.2 Improvement of administration

For lack of time, the administration application is build in a fairly simple way and has only a few functions. In the future, I want to expand this application and to implement new features. I have already implemented a user management system, where every user has his own permissions. This function I want to use and give other people the chance to help me maintain the application.

8.3 Share function

I have been asked by a few persons, if it is also possible to implement a share function. This means, that a user can share his eBooks or libraries to the World Wide Web. This would be a nice feature, that is also possible to implement, but this feature could be also abused by some users to share illegal or hacked books. To avoid this development, I decided not to implement this feature.

8.4 Premium model

As the web services gets more and more users the disk space won't be enough to save the eBooks from all users on Graphitebook. Because this new space would cost money, I'd like to introduce a premium model, when the application gets enough users. Free users can save only a defined quantity of eBooks and libraries, while premium users a higher amount of cloud space. The application could also get some ad banners to finance the server costs.

9 Appendix

9.1 List of figures

9.2 List of references

The Global eBook – A report on market trends and developments – Rüdiger Wischenbart Retrieved January 21, 2014 from http://www.wischenbart.com/upload/Global-Ebook-Report2013_final03.pdf

The eBook market in numbers – Retrieved January 23, 2014 from http://metinmediamath.wordpress.com/2013/12/03/the-ebook-market-in-numbers/