Bilkent University

Department of Computer Engineering

CS 319 Project : *Xtrify*

Group No #10

System Design Report

July 26, 2017

Ali Atlı - 21302442

Seda Gülkesen - 21302403

Atakan Özdemir - 21301134

Usama Saqib - 21500116

Course Instructor: Bora Güngören

This report is submitted to the GitHub in partial fulfilment of the requirements of the Object Oriented Software Engineering Project, course CS319.

Table of Contents

Table of Figures

1.Introduction

1.1. Purpose of the system

Xtrify is a web based note-taking application aiming to record the user's notes and suggest related articles from his/her notes. Different from similar note-taking applications, Xtrify serves as a recommendation system. Once the user save his/her notes, the notes will be extracted into keywords. The user can edit the keywords in order to change the scope of suggestion. These keywords will be used in suggestion stage. Whenever the user wants to find related articles about his/her notes, only thing the user has to do is clicking on advice button. The system requires register in order to go back to his/her history, all the notes and suggested lists are saved into his/her account. The notes are preserved and whenever the user wants to reach them, only need is to login to the system.

1.2. Design Goals

In this section, the design goals which are provided in analysis report will be discussed in order to build proper application.

1.2.1. Reliability

This application is aimed to be bug-free because meeting unexpected terminations could be annoying for the user. To be able to avoid such experiences, the tests will be done for each function of the application.

1.2.2. Modifiability

Our system will be modifiable. Whenever the developer wants to add new features on the application, an update will be easily implemented by the developer. With the help of object oriented design, new classes and objects will be added into the existing code properly.

1.2.3. Adaptability

Comparing with other applications, web based apps are platform independent, they are accessible anywhere, anytime and by a computer with internet connection. Xtrify is a web based application could run on a computer from any operating system. To sum up, the only requirement to use Xtrify is having web browser and internet connection.

1.2.4. Good Documentation

Documentation is one of the important part in software engineering. Team members should give an enough importance of implementation part as well as the documentation. With a proper documentation, requirements are determined and each future of the program is described. As a result, well-documented projects face less failures during the implementation stage. It is also good for providing healthy communication among the team members. While working in a team, documentation is helpful to each member of team.

1.2.5. Well-defined interfaces

Xtrify's user interface is not complicated. It might look like a basic sticky note application, but this interface has been planned intentionally. Since people are struggling to remember things, they want to save his/her notes easily. Well designed, user friendly interface saves the user's time.

1.2.6. Ease of Use

The visitors of the page is impatient and at the first look they mostly decide whether the page meets their expectation or not. Most of the people still prefer to writing their notes by hand. If Xtrify was an ordinary note-taking app, it would be understandable. However, Xtrify is not only note-taking app but also it is regarded as a suggestion system. Xtrify is

the easiest way to reach scholarly journals and articles. Since people from different age groups can easily benefit, simple and easy to understand interface is chosen intentionally. Also, without downloading any extra apps or program, the user can access the application using any web browser and works with internet connection.

2.Software Architecture

2.1. Subsystem Decomposition

For the system three-tier architecture is chosen in order to provide all the classes in the system properly. A three-tier architecture is a software design pattern and a well-established software architecture.[1] Separating the application into tiers provides an ease of maintenance of the code base. Any of the three layers is modified independently so that any change in one tier does not affect other one.

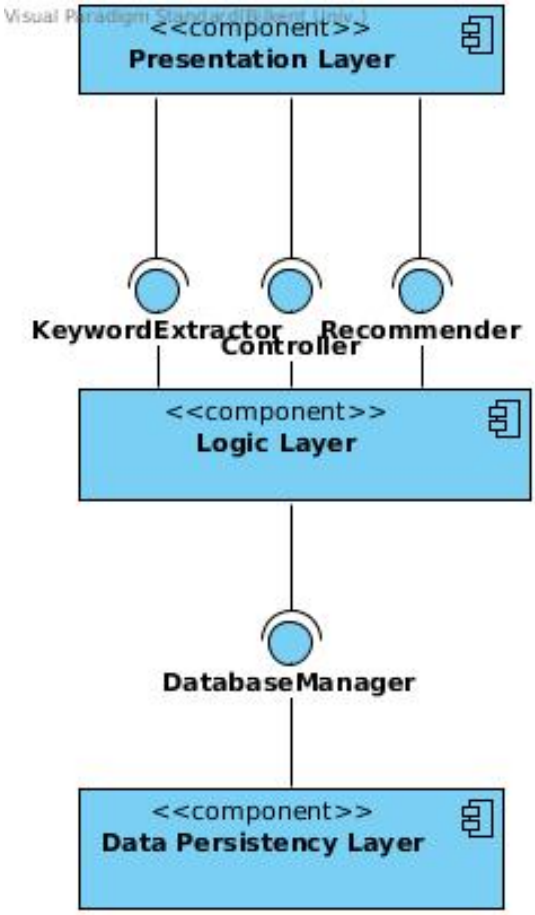


Figure 1: Subsystem Decomposition Diagram

2.1.1. Presentation Layer

This layer has all of the view components of the system. The layer provides application's user interface. The pages are directed to the other pages according to the user's operations.

2.1.2. Application Layer

Application layer is also known as logic tier because it controls the functionality of application.

2.1.3. Data Layer

The data will be kept in an SQLite database, and a database manager will facilitate interaction with the database in order to perform the generic, CREATE UPDATE DELETE operations as per the logic of the application.

2.2. Hardware/Software Mapping

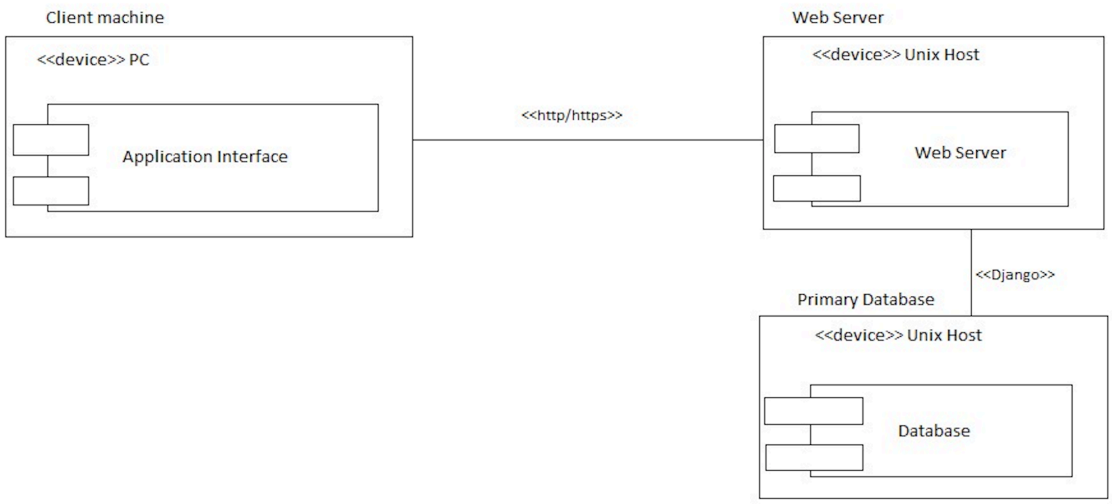


Figure 2: Deployment Diagram for Hardware/Software Mapping

We are dealing with three primary hardware devices. Personal computers where the application interface will be available to the end-users via a browser. Backend server hosting the application itself, and therefore forming the logic layer of the application, in a UNIX environment. Finally a database server, also hosted in a UNIX environment, which will be accessed via the abstraction API provided by Django. The database server is the persistent data storage layer.

2.3. Data Management

Django allows for smooth interfacing of the application with a database backend. The default database used is SQLite, which is also the database backend we have opted for. Connection management can be handled via configuration files, which can be set according to the details of the application. For example, if the application requires constant database access a persistent connection can be set up in order to avoid the overhead of establishing new connections each time, by setting the timeout age to CONN_MAX_AGE. In our case, we do not require a connection of this sort, therefore a more appropriate setting of the age would be 0 (zero), meaning that the connection is closed once the database interaction is completed. This is because interactions with the database are only required when the user explicitly triggers them.

Django automatically gives you a database-abstraction API that lets you create, retrieve, update and delete objects. These can then be used in your python code to manipulate the database entries as the logic of the application desires.

2.4. Access Control and Security

Xtrify requires internet connection and the users need to have an internet browser to access the application. On account of the fact that it is a web based application, nearly every kind of device which has an internet browser can run the application without problem. It just requires an internet connection. When the user is able to run the application with proper internet browser and internet connection, he/she needs to register with a new account or login with an existing account to system with his/her username and a password. The system checks whether the username and password are valid and match then let user in. The user can also change his/her username and password but firstly they need to rewrite their current username and password.

2.5. Boundary Conditions

Initialization

- When the user opens executable file and after s/he access to the internet browser, the application is ready to be used. Register menu screen is shown when the application starts.
- Python and Django framework is used to create pages of the application.
- On the register screen there are three choices: Login, Register and Contact us.
- When the user selects register, he/she needs to write a new username and a password to get a new account, then she/he logins to the system. When the users have any questions or they have in trouble they can contact with the developers via Contact us button.
- When user selects sign in, he/she uses a previously created username and password to login the system.
- On the main page of the application, there are three choices: Home, My Profile and Logout.
- If user selects Home button from the main page, he /she can access his/her previously added notes. In the home page, there are two buttons: save and extract. Save button is for saving the newly added note into the system. Extract button loads keywords page which contains extracted keywords from the corresponding note.
- Keywords page includes advice button which loads suggestions page includes research options of the keywords from the internet database. It also has edit keywords button which loads edit keywords page.
- If user selects my profile from the main page, he/she can access settings which has 2 buttons called change password and change username to change and secure his//her username and password.
- Finally, if the user chooses to logout, he/she will redirect to the login page again.

Termination

- On the main page, the user can choose to logout to exit from the system.
- User can exit with use of “X” button of the browser whenever he/she wants, ongoing process will be lost in this condition.

Failure

- -If user manually closes the browser during the extraction or note taking process, the application will not be able to save ongoing process.
- -The possible problems about power of the system cause the data loss.

3.Subsystem Services

3.1. Services of the Logic Layer

This layer provides the keyword extraction, recommendation and in-site navigation controller capabilities. It is accessed only by the Presentation Layer's classes as dictated by the 3-tier architecture. It uses Data Persistency Layer's DatabaseManager service to validate the login, sign in, sign up actions. Through the DatabaseManager service, the previously recorded notes and other informations are reached and made accessible to the Presentation Layer through Controller service of this subsystem. The navigation between the different pages are handled in this layer as well.

3.2. Services of the Presentation Layer

This component does not have any other higher level layer to provide service. Therefore, one might say it does not offer any services. However, the main loop of the program is within this layer under the hood. It propagates the user interactions on the browser to the lower layers through the services provided by the Logic Layer, as discussed above, and renders the views that are sent by the Controller service.

3.3. Services of the Data Persistency Layer

As our online system promises that the user's notes are stored, the data persistency layer is what makes it possible to keep the data persistent. It is accessed by the Logic Layer.

4.Low-Level Design

4.1. Object Design Trade-Offs

4.1.1. Rapid Development vs Functionality

Due to the fact that we have limited time to develop and implement the application, we need to sacrifice some functionalities of the program to develop faster. At the beginning of the project deciding which functionalities are inevitable and will be implemented primarily is necessary. Firstly, on account of the fact that we use static pages rather than dynamic ones, the user cannot add his/her profile picture into his/her account. Also, if the user forgets his/her password, s/he cannot use his/her e-mail account to reset password. “Forget your password?” function cannot be added because of time issues.

4.1.2. Extendibility vs Functionality

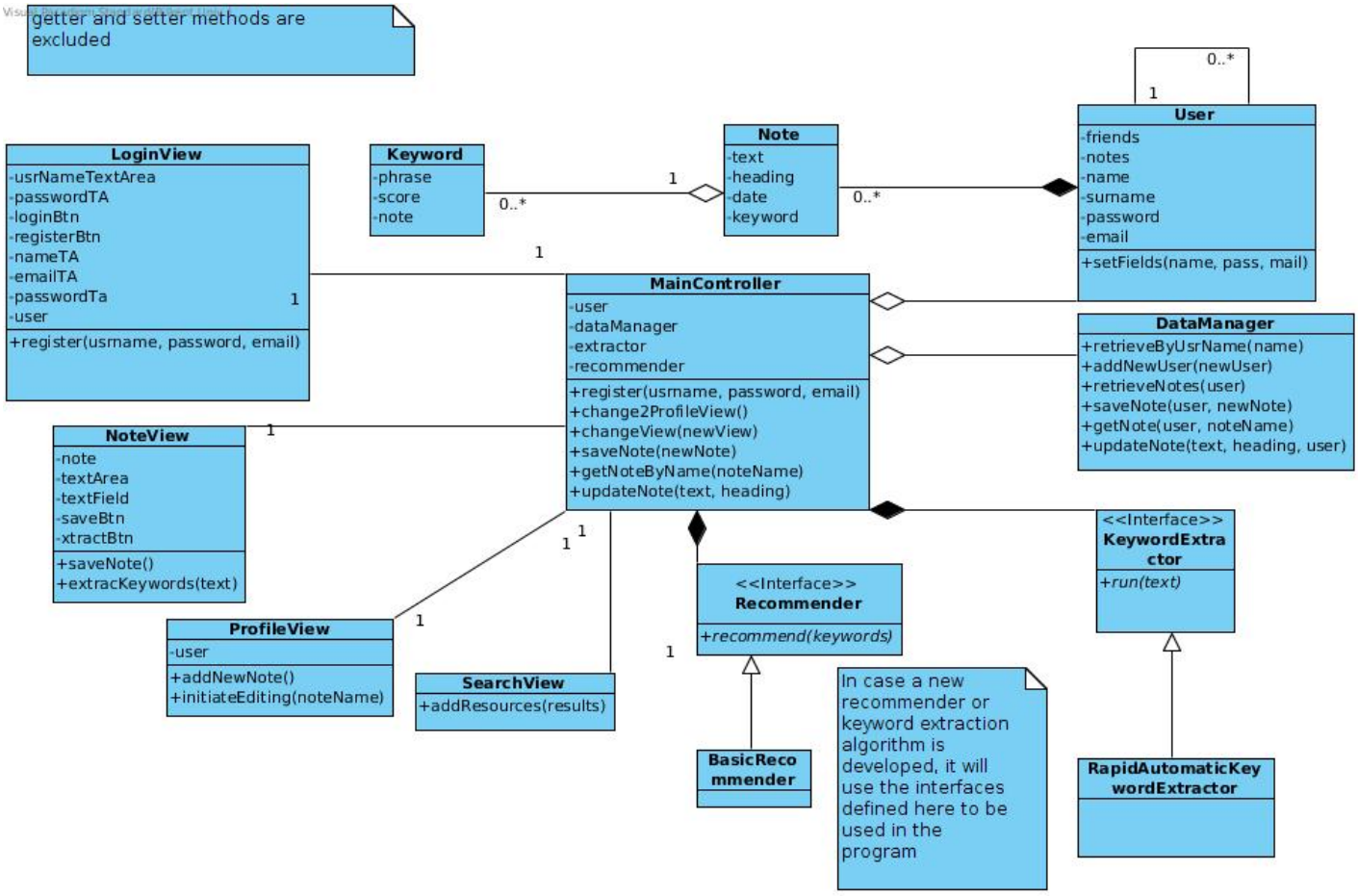
The application at the moment implements only the core functionality. However, in order for it to fulfil all the roles required of a modern note taking application, more features need to be added. Given the time constraints we were not able to add these, therefore, the team’s focus from the beginning was to make the app extensible. This was so that more features could be readily integrated into the current model without changes to the core functionality. This would allow for more rapid development in later periods.

4.1.3. Security vs Performance

Security considerations are ignored in a number of places due to the time constraints and performance requirements. We have decided not to tunnel HTTP requests through SSL as that would require time consuming research. Similar other good practices from a security perspective have been foregone. We have chosen not to encrypt the passwords of the users when storing them in the database, nor perform any salting operations to secure against brute-force attacks, so as to avoid incurring the performance overhead such operations would result in. On top of that the proper implementation of such operations is time consuming and subject to rigorous testing, which would cause the schedule of our project to be extended. Care will be taken while coding to reduce common malpractices which result in command injection vulnerabilities are avoided. However, ensuring that

these practices have been properly followed, require time consuming security audits, and therefore such vulnerabilities will not be protected against.

4.2. Final Object Design



3: Final Class Diagram

4.References

[1] "What is Three-Tier Architecture? - Definition from Techopedia." *Techopedia.com*. N.p., n.d. Web. 25 July 2017.