



Zeeguu Dashboard Architecture

Version 1.0

Authors :

Oliver Holder
Christian Grier Mulvenna
Henry Salas
Tai-Ting Chen
Evi Xhelo
Ai Deng
Jakob Vokac

Clients :

Mircea Lungu
Carlos Paz Rodriguez

Supervisor :

Lars Holdijk

University of Groningen
March 2018

1 Introduction

Zeeguu is an innovative new tool for learning languages. By integrating multiple translating software, such as Google Translate and Microsoft Translator, Zeeguu allows its users to read articles with real time translation at their finger tips and practice their vocabulary with interactive exercises. Right now, Zeeguu supports five different languages to learn (English, Dutch, German, Spanish and French) and three base languages (English, Dutch and Chinese).

Right now, the interaction in Zeeguu is still rather simple. Anyone, who wishes to use it, simply signs up with an access key and inputs some basic information (your name, email address, password, which language you wish to learn and what your base language is). Once the account is created, the user can choose to read articles, practise their vocabulary or look at the words they have learned.

The aim of the Zeeguu Dashboard project (which we abbreviate to ZD) is to extend Zeeguu's functionality in order to help teachers with real world language classrooms. Teachers are already assigning to their students to play with Zeeguu. However as of now, they don't have the ability to analyse their classroom performance when they go to the Zeeguu website. Therefore the solution to this problem has been assigned to us by the Zeeguu authors. Let us briefly get an overview of what comprises zeeguu, then we will ... (christian : maybe here describe how we propose to integrate into this).

2 The Zeeguu architecture

The source for Zeeguu is open and on Github. The source however spans three repositories. See figure 1. These are Zeeguu-API, Zeeguu-Core and Zeeguu-Web. When each of these separate systems are successfully running, you have a copy of the Zeeguu website going. Generally speaking, Zeeguu-Web is for running a web framework, Zeeguu-Core is for making operations on a database and Zeeguu-API provides to both other systems procedures like crawling web articles and assessing language difficulty. According to the Zeeguu-API readme, the API is a thin layer on top of the core.

As seen in figure 1, communication between the Zeeguu-Web and Zeeguu-Core systems does not take place. Rather the the API is responsible for allowing the core to carry out operations like crawling news sites for example. While simultaneously giving Zeeguu-Web access to the database information found in the core. Allow us to paint a somewhat bigger picture by looking at two mind maps in figures 2 and 3. These explore some of the most important ideas in the Zeeguu-Web and the Zeeguu-Core. Note a third mind map for the

API is redundant when it's purpose is to work for the Zeeguu-Web and Zeeguu-Core components. You can learn about it by just looking at these two mind maps.

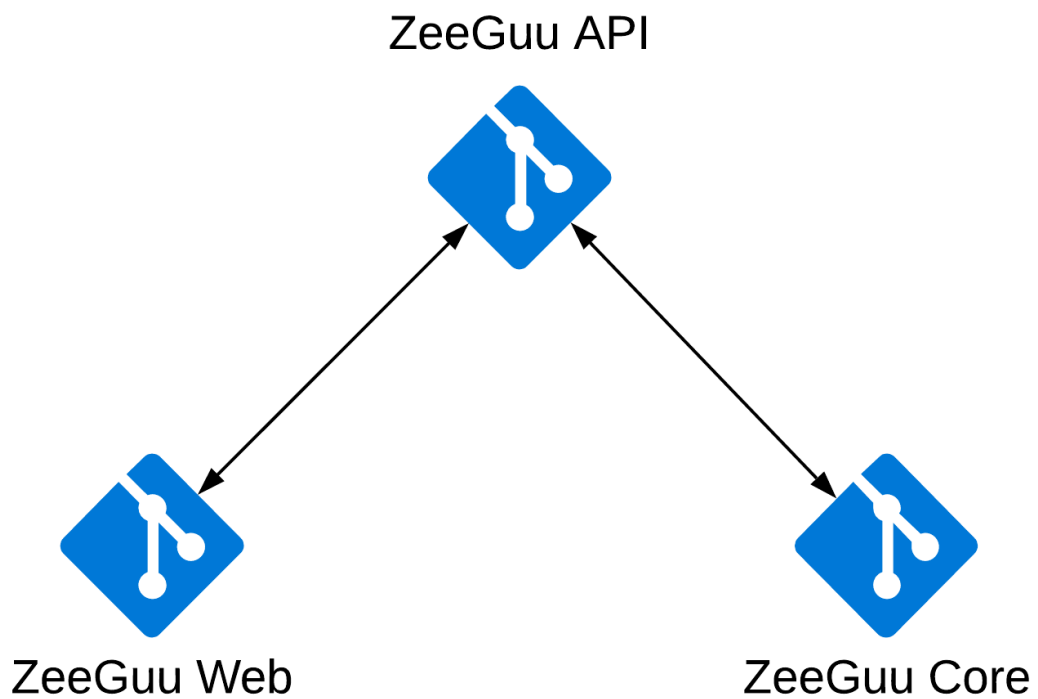


FIGURE 1 – The Zeeguu repositories

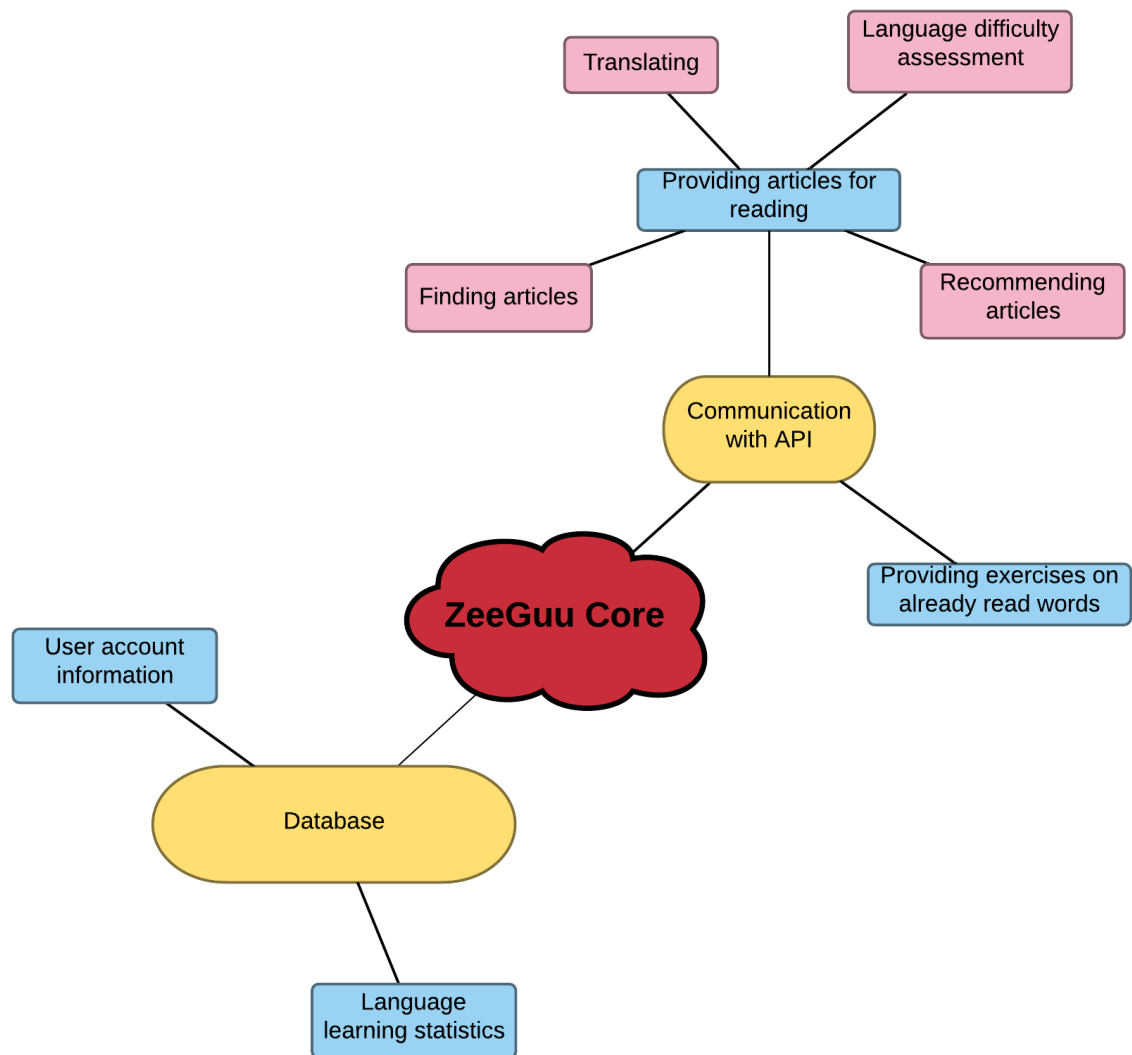


FIGURE 2 – Zeeguu-Core mindmap

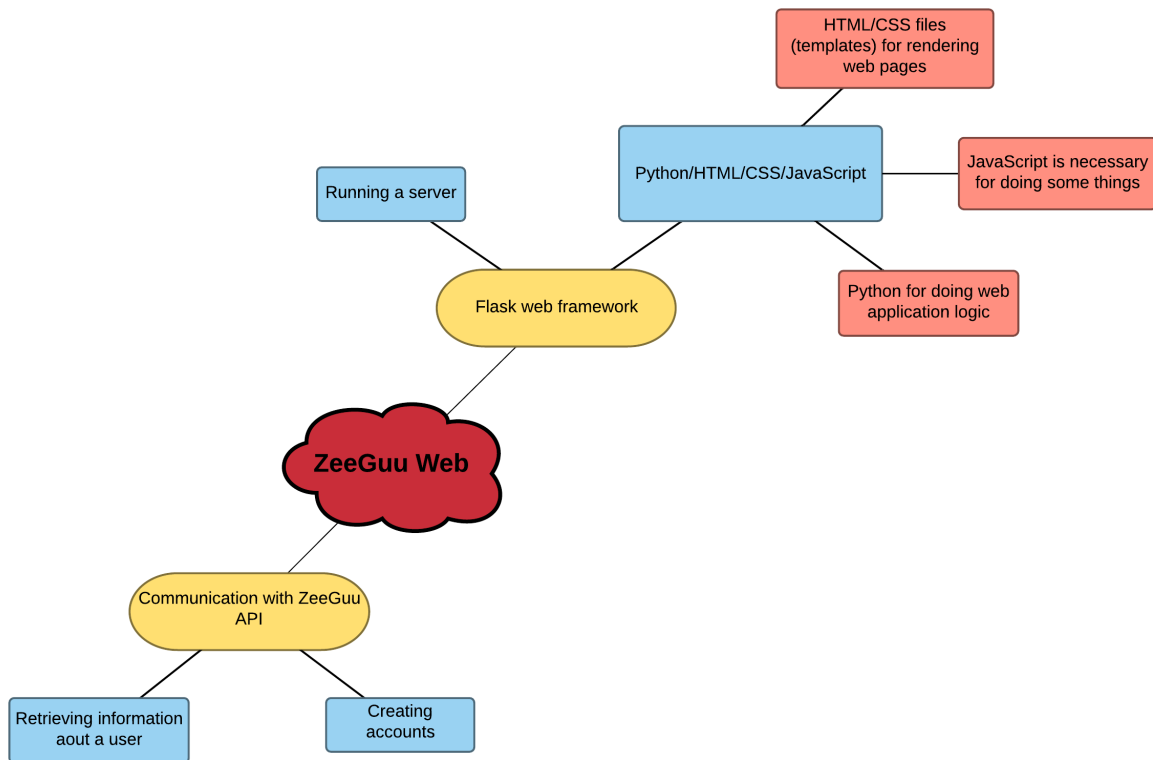
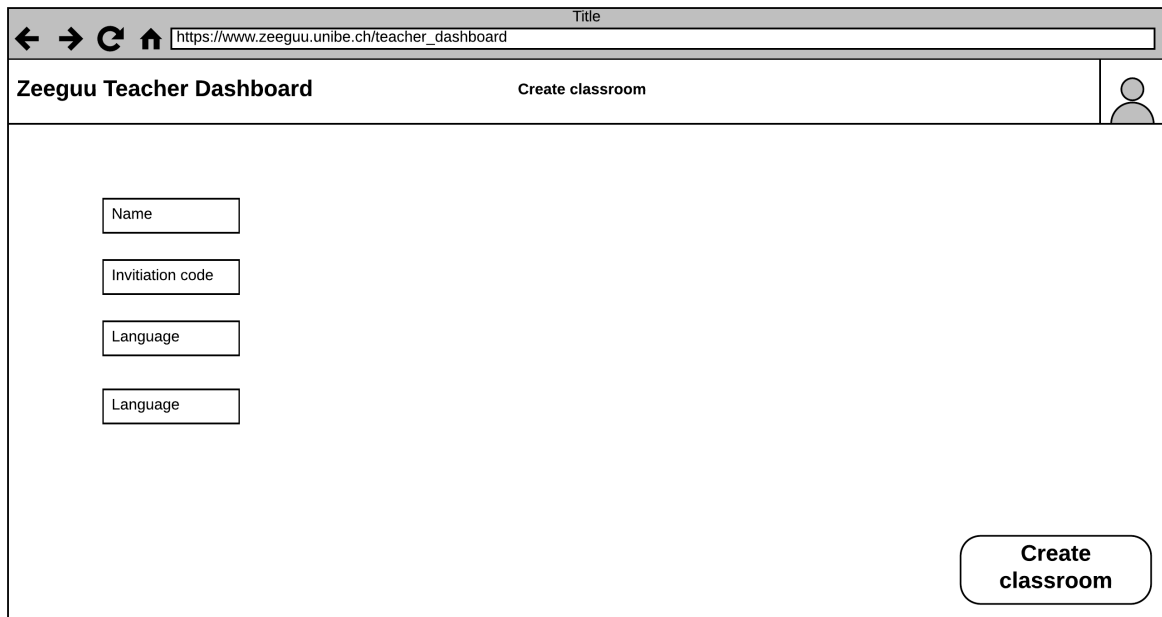


FIGURE 3 – ZeeGuu-Web mindmap

3 Functionality we want

To realise the ZD, for now we simply want three main web pages. One is for allowing a teacher to create a new classroom, another for a teacher to see all the classes they have already made and the last for investigating student progress within a class. These concepts have mock-up pictures respectively listed in figures 4, 5 and 6. These pictures don't represent our design goals but rather describe the functionality we want right now.



This mock-up shows the 'Create classroom' interface of the Zeeguu Teacher Dashboard. The browser's address bar displays 'https://www.zeeguu.unibe.ch/teacher_dashboard'. The page header includes the title 'Zeeguu Teacher Dashboard', a 'Create classroom' link, and a user profile icon. The main content area contains four input fields: 'Name', 'Invitation code', and two 'Language' fields. A 'Create classroom' button is located in the bottom right corner.

Zeeguu Teacher Dashboard

Create classroom

Name

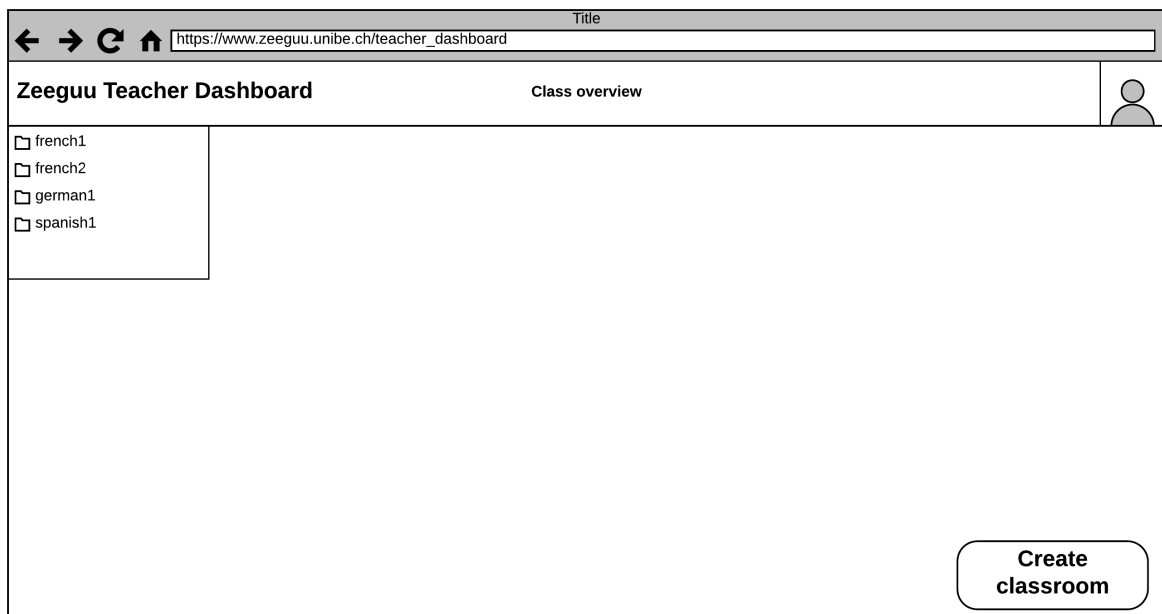
Invitation code

Language

Language

Create classroom

FIGURE 4 – Create classroom mock-up



This mock-up shows the 'Class overview' interface of the Zeeguu Teacher Dashboard. The browser's address bar displays 'https://www.zeeguu.unibe.ch/teacher_dashboard'. The page header includes the title 'Zeeguu Teacher Dashboard', a 'Class overview' link, and a user profile icon. The main content area features a sidebar with a list of classes: 'french1', 'french2', 'german1', and 'spanish1'. A 'Create classroom' button is located in the bottom right corner.

Zeeguu Teacher Dashboard

Class overview

french1

french2

german1

spanish1

Create classroom

FIGURE 5 – Class overview mock-up

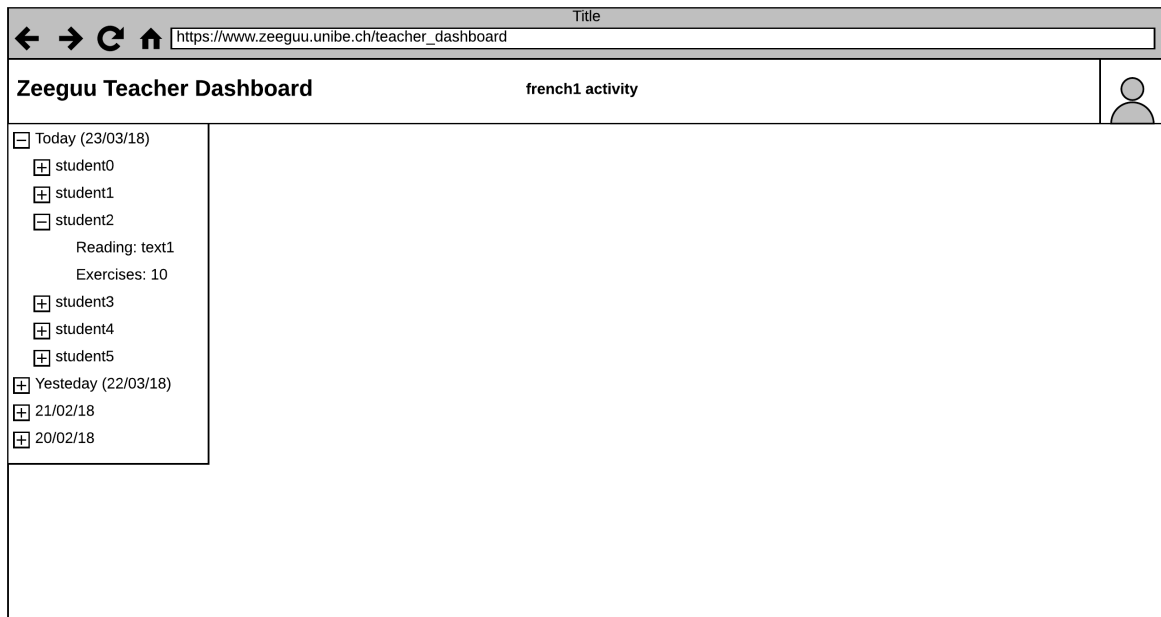


FIGURE 6 – Class activity mock-up

4 Implementation

To know how to make this functionality happen, we have categorised different elements of the functionality to better the understanding of what needs to be done. These categories are front-end and back-end.

The front-end work comprises :

- Putting together CSS/HTML code to design the aesthetics for each web page
- Connecting the routing logic in the python-flask-framework to things like buttons outlined by the CSS/HTML
- Importing relevant data from the API using requests.
- Posting to the API for create class and login

The back-end work encompasses :

- Updating the database to support the creation of a new classroom
- Updating the API to support requests from Zeeguu-Web to Zeeguu-Core to create new classrooms
- Updating the API to support requests from Zeeguu-Web to Zeeguu-Core to get information on classrooms and students

In implementing these requirements, we had to make some concrete decisions.

For the front-end implementation :

- All 'GET' and 'POST' methods are called through specific functions written to interact with the API. This is done to reduce code duplication and unnecessary errors when dealing with issues such as including a 'sessionID'.
- All data gathering functions are split from the routing because we want forward integration with future API changes to be facilitated.
- Sessions has been implemented so a user on the website cannot request for data that the API will reject. Instead it redirects you to the log in page or a permissions denied page. This permissions handling has been split into a separate python file for code clarity.

Back-end implementation choices :

- Every function that returns data is protected unless given a valid 'session' to access it. This is handled by permissions checking functions.
- All data sent from API to WEB is sent in JavaScript Object Notation format.
- All data interactions have been optimised to reduce 'GET' requests to the API.
- Permission checking functions have to included to facilitate the website permissions enforcement.

5 Technology stack

1. HTML - Used for designing the page
CSS - Used for styling the web-page.
JQuery - Used for easier web-page design.
2. Python - Used for the back-end of the project.
Flask - Used in Python for developing websites.
Jinja2 - Used for dynamically constructing web-pages in HTML.
WTForms - Used for easy form creation and usage in HTML.
Bootstrap - Provides templates for better looking webpages.
SQLAlchemy - Integrates server queries into python.
3. MySQL - The framework on which the project server is built on.

6 Team organisation

The group project team is made up of seven people. We have put ourselves into three teams the front-end team, the middle-back-end team and the back-end (API) team. Each team is listed below, including each team member.

Front-end team :

- Henry Salas
- Evi Xhelo
- Ai Deng

Middle-back-end team :

- Oliver Holder
- Christian Grier Mulvenna
- Tai-Ting Chen

Back-end team :

- Oliver Holder

7 Change Log

Created requirements document on the 27th at 5pm.

When	Where	What
Mar 17th, 20 :00	The document	Draft made for the structure ; Intro written
Mar 22nd, 11 :00	The Zeeguu architecture	Images included ; Most of the text written
Mar 23rd, 11 :00	The Zeeguu architecture	Picture fixing
Mar 24th, 15 :30	Functionality we want ; Im- plementation	Some text and images made
Mar 24th, 15 :30	Technology stack	Text made
Mar 25th, 15 :00	Everywhere	Last touches before the end of the second sprint