

TELECOM NANCY

UNIVERSITÉ DE LORRAINE

ENGLISH PROJECT

Anglofun

Auteurs :
Tom HOHLER
Vincent ALBERT

May 1, 2016

Contents

1 Introduction

Présentation du sujet

1.1 A brief presentation of the project	1
1.2 Specifications	1

2 User Manual

2.1 The basics: How to create an account and to log in.	2
2.2 The front page for connected users: The Ali Baba's cave of lessons.	2
2.3 Now the real thing: How to try a lesson.	3
2.4 Want to see your progression? You can find everything in your account.	4
2.5 To go further: Let us choose how to train your english!	4
2.6 Now, you are the teacher: Post a new lesson, and help other students to improve themselves!	5

3 La mise en œuvre technique du projet

3.1 L'organisation du projet	6
3.2 les technologies mises en pratique	6
3.3 Les attentes atteintes et ce qui aurait pu être fait	8

4 Conclusion

Références

1 Introduction

1.1 A brief presentation of the project

There are lots of applications on the Internet to help people improve their english. But we wanted to do it in a really pedagogically way, so that the learner won't give up after a few weeks. And so, to motivate him and push him to learn without counting the time, we created a website designed to teach "students" users with interesting stories. You can access the website though this URL: <http://1.anglofun-1291.appspot.com>

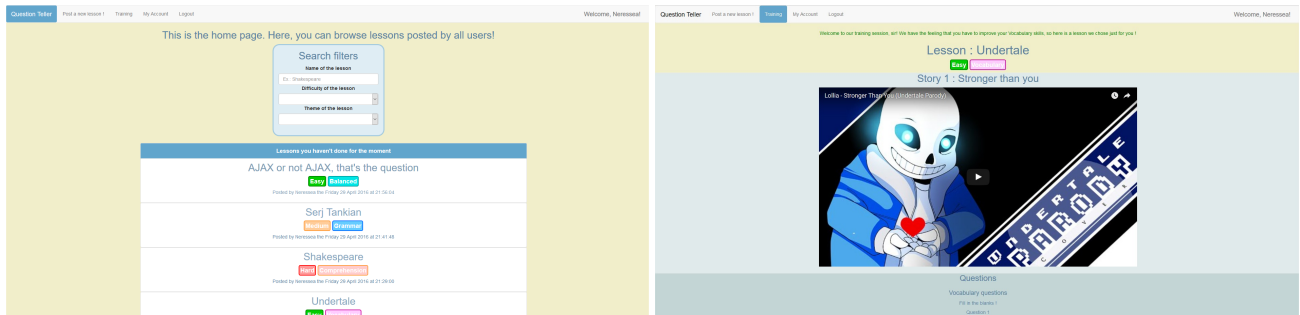


Figure 1: Snapshots of the website

1.2 Specifications

So our objectives were to allow users to try lessons to improve themselves, each lesson being composed by several stories and questions related to it. But we didn't want to create a huge and static database of stories from the beginning. We thought it would be more entertaining if all users could create their own stories and send it to other users. But that's not all. The main feature of our application is certainly the "Training program" proposed by the server, in which the website trains the user by sending him lesson. To perform the best user experience, we also wanted the website to be responsive, so the user can either use it on a computer or a tablet.

Now, we will see in details these different features and after that, we will present you their technical side.

2 User Manual

We will now see all the possibilities of our web application in details. In order to do that, we will follow the classical path of an ordinary user.

2.1 The basics: How to create an account and to log in.

When your firstly go on the homepage of the website, you meet a form to log in. As in other applications, you just have to put your credentials to log in. If you want to create a new account, you just have to click on the corresponding button. On the new page, you juste choose your credentials and send them.

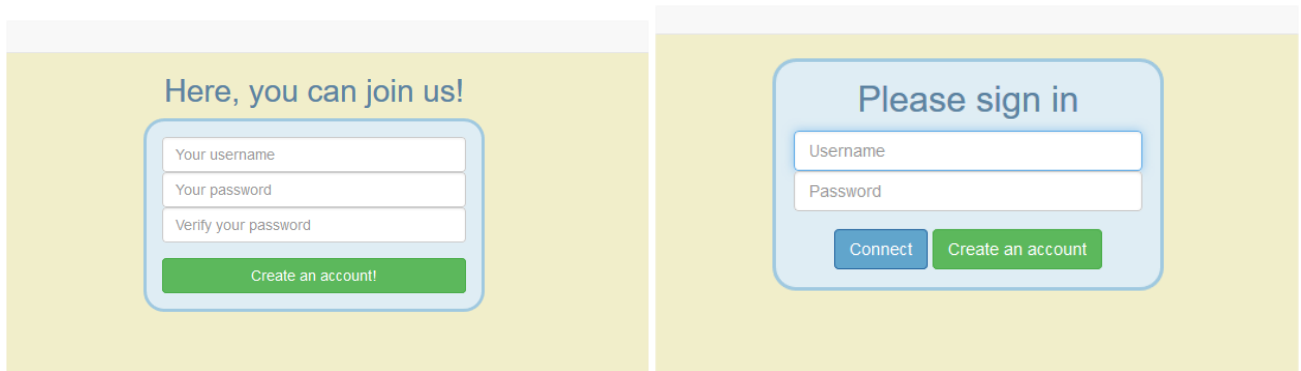


Figure 2: Login and signin

Now that we passed through all the formalities, we can go on the real features of our application!

2.2 The front page for connected users: The Ali Baba's cave of lessons.

Now that you are connected to your session, the front page is not the same anymore. All the lessons posted by other users that you haven't done are listed here! You can browse to find what you want. The lessons are ordered by creation date, from newer to older.

For each lesson of the list, you can see its title, its difficulty, its theme, the name of the user who posted it and the creation date. The theme of the lesson is the dominant theme according to prevalence of questions. There are three kind of themes : Vocabulary, Grammar and Comprehension. Balanced lessons don't have any dominant theme, there are the same amount of each type of questions.

You want something specific? Search filters are here to help you find what you really want! You can search lessons by name, by difficulty and dominant. Of course, you can user several filters at the same time.

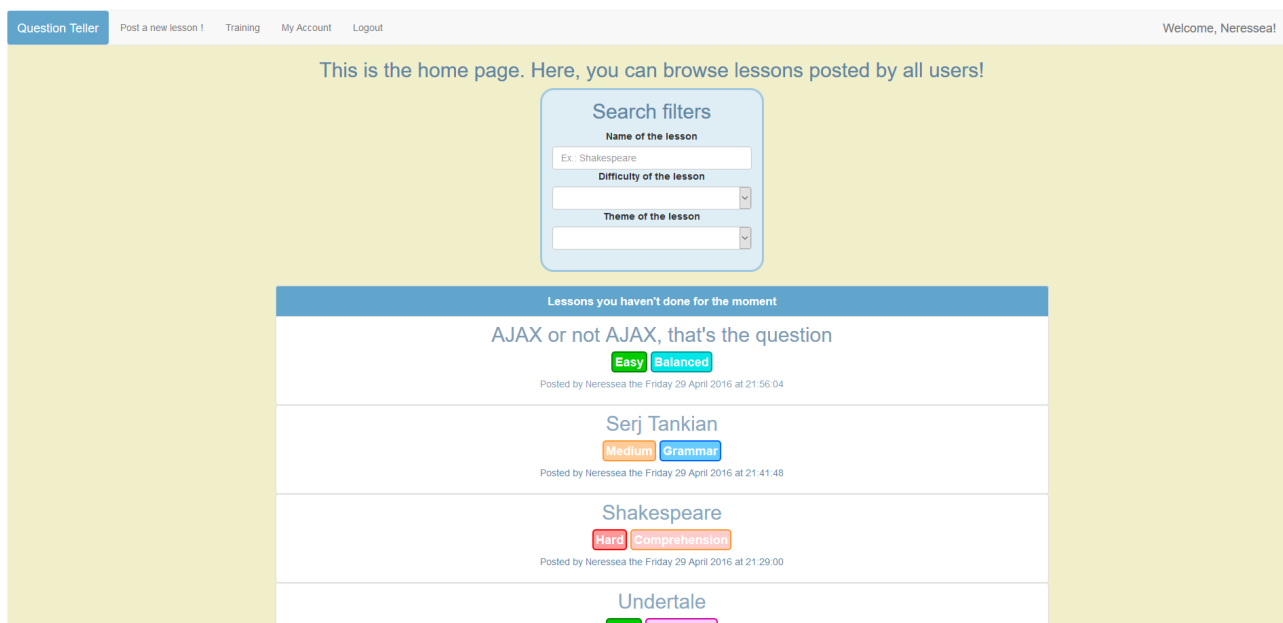


Figure 3: Front page and search filters

Now the front page is pretty and all, but we think that you don't want to stay here forever. So you can navigate through sections thanks to the navigation section, or click on a lesson to try it out.

2.3 Now the real thing: How to try a lesson.

Now that you chose your lesson, the time has come to face it! A lesson can have several stories. A story can be either a text or a youtube video. For each story, the three kind of questions grouped in sections : the first is for vocabulary questions, the second is for grammatical questions and the third is for comprehension questions.

In each of these sections, you can have several questions. The vocabulary questions are fill-in-the-blank exercises related to the story, grammatical questions are 4-choices MCQ and finally comprehension questions are direct questions where only one answer is accepted.

When you filled all the answers you know the answer, you can check it. The website will show you the correction of the lesson. If your answer was correct, it is shown in green. If it is wrong, it is shown in red and next to it there is the correct answer in green. At the top of the lesson, you have your statistics for this lesson : the percentage of correct answer for each type of questions. These statistics will be add to your general stats in your account section.

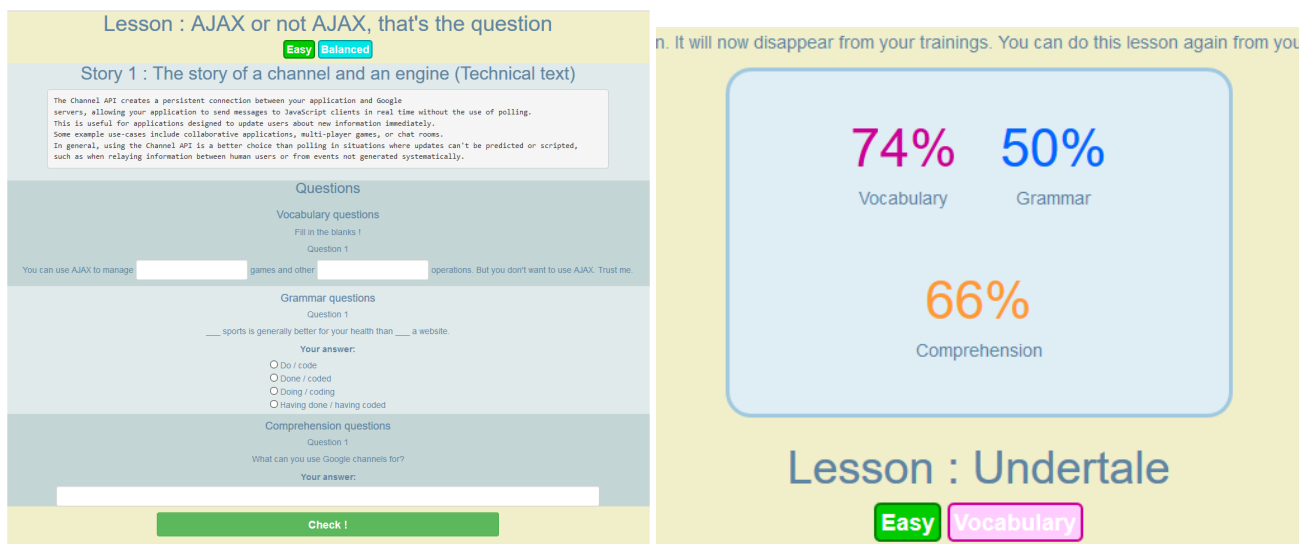


Figure 4: When you try out a lesson

2.4 Want to see your progression? You can find everything in your account.

If you want to track your progress - and if you really want to progress you are likely to do it - you can see it from your account section. Here, you can see at the top of the page your global stats for each type of question. This can help you to know your weaknesses and strengths. Below that, you have the list of all lessons you have already done. These won't show up again in the front page nor in the training program, but you can still access them from here and try them again. But now, what is precisely the training program?

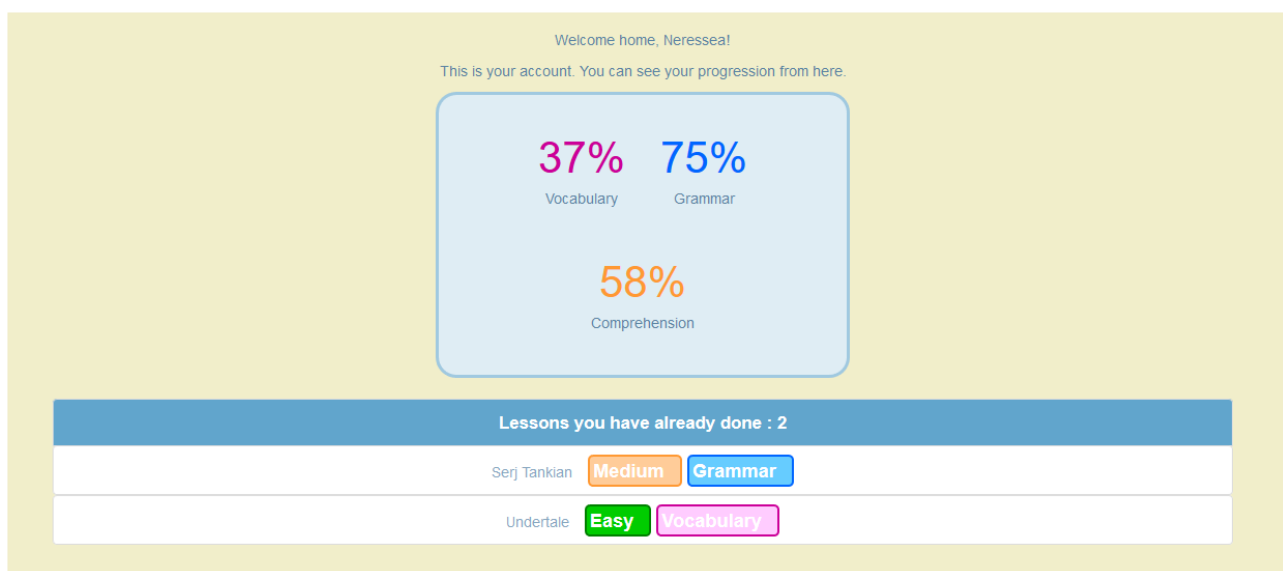


Figure 5: Your account page

2.5 To go further: Let us choose how to train your english!

So, if you go on the "Training" section in the navigation bar, the server will challenge you with a lesson of its own. But it won't choose it from nowhere ; in-

deed, the server will check what your weaknesses are, and will search for a lesson that can help you to improve your skills in this domain, and that you haven't already done. But if it does not find a lesson that you haven't done, it will at this moment send you a random lesson. And if you have already done all the lessons of the database, it will just send you an excuse message. However, if you already did everything, I think you already are skilled enough!

So now you know about everything as a student user. But what if you want to send your own lesson?

2.6 Now, you are the teacher: Post a new lesson, and help other students to improve themselves!

If you want to help others, you can also change hats and go to your teacher hat. To do this, you just have to go in the section "Post a new lesson!". Here, you can fill in all the required fields concerning the lesson: its title and the difficulty you think it has.

After that, you can add new stories to the lesson with the related link and fill in them. For each story, you have to choose a title and set the content of the story. You can also choose the type of the content (text or youtube video). At the moment you change the content of the story, you have the preview of its display. It is really helpful if you want to listen to the video you want to set while adding questions. You won't have to change of page every time you want to set pause.

After that, you can fill questions. You can add questions to each section (Voca, Grammar, Comprehension) as you wish. Now you just have to fill fields for each question. For vocabulary related questions, you just write the text you want. And to indicate us which words you want to be replaced by a hole, you just surround it with the markers [hole][hole]. For instance : "Yesterday, I bought [hole]bread[/hole]". In this example, the word "bread" will be replaced by a blank for the user. Regarding the grammar questions, they are all MCQ where you fill in the question and the 4 different answers. The first answer will be the correct one. When it is displayed to the user trying it, the answers are shuffled, of course. And finally, the last kind of question : the comprehension ones. Here, you just have to give us the question and the answer. You can note that the answers of the user must exactly the same for fill-in-the-blank and direct questions. For instance, if the answer you gave is "The answer" when creating the question, "answer" won't be considered as a correct answer. However, it doesn't matter if the answer is uppercased or lowercased. In our example, the answer "THE ANswER" will be considered as correct.

Now that you filled in all the fields, you can send us your lesson! And don't worry: the website is kind enough to tell you if you forgot fields. *Now that we saw everything that is possible with our application, we will see how it is done in the code!*

3 La mise en œuvre technique du projet

Nous rappelons que le site web peut-être testé depuis le lien suivant : <http://1.anglofun-1291.appspot.com>

3.1 L'organisation du projet

Afin de travailler dans les meilleures conditions possibles, nous avons utilisé le gestionnaire de version **git** qui permet de gérer de manière approfondie le versionnage du projet et de travailler simultanément plus facilement. Le projet se trouve sur le dépôt public <https://github.com/Neressea/EnglishProject>

3.2 les technologies mises en pratique

Afin de décrire au mieux les technologies employées dans notre projet, nous allons distinguer ses deux principales parties : la partie back-end et la partie front-end.

Côté back-end. Du côté back-end, nous avons utilisé du python afin de coder le serveur web à l'aide du framework fourni par le Google App Engine qui permet entre autre de gérer les routes ainsi que le contenu des requêtes, et qui permet d'utiliser efficacement l'Active Record dans la gestion de la base de données.

Nous avons aussi utilisé le gestionnaire de modèles Jinja2 permettant de générer des pages HTML à partir de "modèles" contenant non seulement le code HTML, mais aussi du code python (la génération des pages HTML devient ainsi plus flexible).

Enfin, nous avons créé la base de données selon le modèle relationnel suivant :

USER

- isAdmin : boolean
- username : String
- password : String
- grade_vocabulary : int
- grade_grammar : int
- grade_comprehension : int
- lessons_done : List(int)

Nous pouvons noter que l'attribut isAdmin, bien que précisé dans la base de données, n'est pas utilisé dans le reste du code. En effet nous n'avons pas eu le temps d'ajouter les privilèges administrateurs

au site web.

Le mot de passe quant à lui est hashé avec la fonction hmac qui prend en paramètre un sel afin de rendre le hashage plus sécurisé. Il est aussi important de noter que cette fonction de hashage prend toujours le même temps, peu importe les capacités physiques de la machine ce qui permet d'éviter les bruteforce.

Enfin, la liste des leçons faites est une liste des id des leçons.

LESSON

- created_by : String
- difficulty : String in [Easy, Medium, Hard]
- title : String
- created : Date
- last_modified : Date
- dominant : String in [Vocabulary, Grammar, Comprehension, Balanced]

La difficulté et le thème majeur de la leçon sont des chaînes de caractères dont les valeurs doivent être comprises dans une liste de valeur acceptée.

STORY

- id_lesson : int
- num_story : int
- title : String
- type_of_story in [Text, Video]
- text : String
- questions_vocabulary : List(String)
- questions_grammar : List(String)
- questions_comprehension : List(String)

Le numéro de l'histoire correspond au numéro de l'histoire dans la leçon. Le type correspond au type du contenu et ne peut être que du texte ou une vidéo. Les questions sont des listes de chaînes. Chaque élément de la liste correspond à une question. La chaîne est formatée de la manière suivante : "question/bonne_réponse/mauvaise_réponse/..."

Côté front-end. Pour le front-end, nous avons utilisés les technologies classiques (CSS, HTML, javascript). Mais afin d'être plus efficace et de proposer une expérience plus poussée à l'utilisateur, nous avons utilisés des frameworks pour ces technologies. Pour le CSS, nous avons utilisés le framework bootstrap qui nous a permis de réaliser un design responsive plus rapidement. Et étant donné toutes les modifications effectuées sur le DOM de la page HTML (génération du formulaire en javascript, ajout dans le formulaire lors des ajouts d'histoires / questions, changement lors de la preview, ainsi que filtres de recherches) nous avons décidé d'utiliser jQuery afin de manipuler les sélecteurs CSS plus facilement. Du côté du front-end, nous avons rencontré quelques problèmes dans l'affichage de la vidéo (lors de la preview ou de l'affichage de la leçon). En effet, une connaissance nous a fait remarqué, après utilisation de notre site web sur son smartphone, qu'il ne pouvait ni voir de vidéo en preview ni en uploader. Nous nous sommes alors rendus compte qu'il existe 4 types de liens Youtube : 1 pour le site classique, un pour le site en version mobile, un pour l'application smartphone et enfin un pour les iframe. Dans notre cas, nous transformions les liens du site classique en lien pour iframe par remplacement de string, mais nous ne prenions pas encore les deux autres liens en compte. Nous les avons donc ajoutés.

3.3 Les attentes atteintes et ce qui aurait pu être fait

Toutes les fonctionnalités initiales que nous désirions implémentées ont été faites. On peut cependant noter que certaines fonctionnalités que nous avons commencé à implémentées n'ont pas pu être terminées.

Les comptes administrateurs. Bien que déjà implémenté dans la base de données, nous n'avons pas eu le temps de mettre en place les privilèges accordés aux administrateurs (édition et suppression de leçons, suppression de comptes utilisateurs).

Droits utilisateurs. En l'état, un utilisateur ayant posté une leçon ne peut pas l'éditer ultérieurement. Nous aurions pu offrir cette possibilité.

Le choix du type de question. Nous aurions pu proposer à l'utilisateur de choisir le type de question (texte à trous, QCM, question directe) désirée au lieu de décider ce type en fonction du thème de la question (vocabulaire, grammaire, compréhension).

Un formulaire plus adaptatif. Dans le formulaire, nous permettons d'ajouter des champs, mais pas de les supprimer. Cette option pourrait être intéressante si l'utilisateur se trompe dans l'ajout d'une question ou d'une histoire.

Des réponses plus flexibles. Les réponses données pour les questions de compréhension doivent pour l'instant être complètement identiques à la réponse entrée par l'enseignant. Les leçons seraient plus agréables pour l'élève si elles pouvaient admettre des réponses différentes ("Obama" à la place de Barack Obama, ou même des réponses très différentes mais toutes deux justes : "The president of the United States" au lieu de "Barack Obama").

Une utilisation de la courbe d'oubli. Enfin, nous aurions pu mettre à escient la "courbe d'oubli" afin de réduire les notes de l'utilisateur au cours du temps, pour que celui-ci finisse de nouveau par avoir des leçons sur les thèmes où il était déjà bon.

4 Conclusion

So, we saw in the previous part that we didn't manage to implement all the features we wanted to do (like administrators rights, usage of the forgetting curve, and more flexible forms). However, we succeeded in doing the key features of this project and we think that it is sufficient to already provide a nice experience to the user.

Besides of what we have done, this project allowed us to learn new technologies like jQuery and bootstrap, and to improve our skills in the others we already knew like python, Jinja2, and straight JTML, CSS and javascript. We also had to put ourselves in the place of the user, and it was really a complicated exercise to think about everything. To have several opinions, we also asked our acquaintances to try it and tell us what they thought about it.

As a conclusion, we think that this project was a great opportunity to explore new programming tools, and to create a real-size website during several months.

Annex

`https://cloud.google.com/appengine/docs/python/refdocs/`
We used python for the server.

`http://jinja.pocoo.org/docs/dev/api/`
We used a template manager to generate flexible HTML.

`http://getbootstrap.com/css/`
We used bootstrap to manipulate CSS.

`http://www.w3schools.com/jsref/`
We used javascript to manipulate the HTML from the client side.

`http://texdoc.net/texmf-dist/doc/latex/latex2e-help-texinfo/latex2e.pdf`
And finally, we used \LaTeX to write this report!

We also want to thank everybody that used our application and allowed us to improve it!