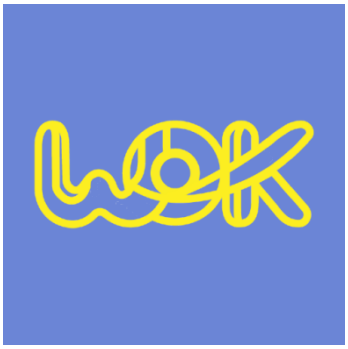


WOK – User Interaction in a Crowdsourcing Application

Lourenço Francisco Costa Palma

Thesis to obtain the Master of Science Degree in

Information Systems and Software Engineering

Supervisors: Prof. Duarte Nuno Jardim Nunes,

Prof. Sónia Isabel Ferreira do Santos Rafael

Examination Committee

Chairperson: Prof. Alexandre Paulo Lourenço Francisco

Supervisor: Prof. Duarte Nuno Jardim Nunes

Member of the Committee: Prof. Augusto Emanuel Abreu Esteves

November 2019

Acknowledgments

Firstly I would like to thank my team members David Lopes, Márcia Marranita and Miguel Garrido, as this project wouldn't be completed without them.

I would also like to thank my supervisor, professor Nuno Nunes from Instituto Superior Técnico, and my co-supervisor, assistant professor Sónia Rafael from Faculdade de Belas Artes da Universidade de Lisboa for all the valuable advice, availability and guidance along the process. Without them it would be impossible to complete this project and make our solution, WOK, come to fruition.

I would like to thank Instituto Superior Técnico for the possibility of performing this project.

I would like to express my gratitude towards Unbabel for providing us the tools and space for executing this project.

I would also like to thank Carlos Silva, Diogo Lopes, Miguel Pinto, and Ricardo Monteiro from the other IST SCOPE team for helping us during the user testing phase and for their constructive criticism and input.

I would like to thank Pedro Oliveira for his help during the user research and João Ferreira for his advice and feedback during the design process.

Lastly, I would like to express my gratitude towards my family and friends for all their help and support.

Abstract

The multidisciplinary project IST SCOPE WOK, made in partnership with Unbabel, consists of a Crowdsourcing application for the Facebook Messenger Platform based on the Wisdom of the Crowd concept. Users perform tasks supplied by a task provider bot represented in the form of Co-Workers of the user. Co-Workers have different personalities and interact with the users providing them tasks and incentivizing them to perform said tasks, with the goal of increasing user productivity. Tasks can have two types: Quality Estimation or Categorization. In Quality Estimation tasks the users must evaluate the correctness of a short text with a yes or no answer. In Categorization tasks, the users must assign a category and sub-category to a short text. Tasks are performed in *Webviews*, embed browser engines inside the Facebook Messenger Platform.

Keywords

Bot, Client, Co-Worker, Crowdsourcing, Task, User, Webview

Resumo

O projeto multidisciplinar IST SCOPE WOK, realizado em parceria com a Unbabel, consiste numa aplicação de Crowdsourcing para a Plataforma Facebook Messenger, baseada no conceito de Wisdom of the Crow (Sabedoria das Multidões). Os utilizadores realizam tarefas disponibilizadas por um bot provedor de tarefas, representado na forma de Colegas (Co-Workers) do utilizador. Os Colegas têm diferentes personalidades e interagem com os utilizadores, facultando-lhes tarefas e incentivando-os a realizar ditas tarefas, com o objetivo de aumentar a produtividade do utilizador. As tarefas podem ser de dois tipos: Estimação de Qualidade (Quality Estimation) ou Categorização (Categorization). Nas tarefas de Estimação de Qualidade, o utilizador deve avaliar se um pequeno texto está correto com respostas de sim ou não. Nas tarefas de Categorização os utilizadores devem atribuir categorias e subcategorias a um pequeno texto. As tarefas são realizadas em *Webviews*, motores de pesquisa embebidos dentro da Plataforma Facebook Messenger.

Palavras-Chave

Bot, Cliente, Colega, Crowdsourcing, Tarefa, Utilizador, Webview

Table of Contents

List of Figures

Glossary

Bot – also known as web robot is a software application that runs automated tasks (scripts) over the Internet.

Capstone – a multifaceted assignment for students during their final year of high school or at the end of an academic program. Most are long term projects that conclude with a final product and presentation.

Co-Worker – a virtual colleague of the user.

Crowdsourcing – is a sourcing model in which individuals or organizations obtain goods or services from a large, open and rapidly-evolving group of Internet users.

Framework – a concrete or conceptual platform where common code with generic functionality can be selectively specialized or overridden by developers or users. Takes the form of libraries, where a well-defined API is reusable within the software under development.

Handler – software routine that performs a particular task.

Micro-services – a software development technique that structures an application as a collection of loosely coupled services.

Payload – part of the transmitted data sent with a message.

Persistent Menu – A menu always available to the user, containing top-level actions the user can enact at any point.

Persona – fictional characters created based upon research in order to represent the different user types that might use a specific service, product, site or brand. Personas are created to help understand the users' needs, experiences, behaviors, and goals.

Service – software that performs automated tasks, responds to hardware events or listens for data requests from other software.

Webview – A browser engine contained within an application.

Wisdom of the Crowd – is the collective opinion of a group of individuals rather than that of a single expert.

Acronyms

AI – Artificial Intelligence

API – Application Programming Interface

CSS – Cascading Style Sheets

CSV – Comma-separated Values

DOM – Document Object Model

GIF – Graphics Interchange Format

HTML – HyperText Markup Language

IEM – Iowa Electronic Markets

IM – Instant Messaging

IRC – Internet Relay Chat

IST – Instituto Superior Técnico

JSON – JavaScript Object Notation

ML – Machine Learning

TSV – Tab-separated Values

URL – Uniform Resource Locator

WOK – Wisdom Of the Krowd

1 Introduction

This chapter introduces the project, explaining the model approach, concepts, motivation and goals while introducing the team.

1.1 Capstone Model

This project followed a Capstone Model [1] approach, a diverse project that serves as the genesis for this project report, culminating in a rich academic and intellectual experience for the students involved. The goal of this model is to make students experience the needs and challenges of the labor market. This is accomplished by using an interdisciplinary approach to solving complex problems in highly competitive global markets.

The SCOPE project by Instituto Superior Técnico (IST) [2], having the Capstone Model as a base, attempts to introduce an innovative model based on interdisciplinary collaboration among different engineering fields at IST, enabling dissertation and project students to collaborate in small teams to solve real problems set by companies while also being integrated in those companies environment.

1.2 Motivation

This project and its concept were proposed by *Unbabel* [3], a company that provides AI-powered human translation software. The concept consists of a Crowdsourcing [4][5] task solving service where human users would perform several tasks submitted by clients of this service; once the tasks are completed, the clients would obtain the completed tasks. The idea originates from the Wisdom of Crowd [6][7] effect, that serves as the baseline for this project, hence the name of the project is *WOK*.

1.3 Teams

This project was performed in the context of the IST pedagogical innovation and was developed by a team of two Information Systems and Computer Engineering students (myself included) from IST, a Physics Engineering student from IST and a Communication Design and New Media student from Faculdade de Belas Artes da Universidade de Lisboa (FBAUL) [8], observable in figure 1.1.

Figure 1.1: IST SCOPE Team with designated roles.

1.4 Goals

The goal for this project is to perform user research, create and implement the task solving service and develop branding for our product.

In this project report, I will describe my contributions to the project, as well as the implementation, improvements, architecture, and features, while highlighting some important decisions made along this process.

2 State of the Art

This chapter explains the current state of Crowdsourcing applications, explaining the different Crowdsourcing strategies, Micro tasks, and Chatbot interactions.

2.1 Crowdsourcing

The main concept for this project is Crowdsourcing [4][5][46][53][54], a sourcing model where individuals or organizations use the contributions of large, open and rapidly-evolving groups of internet users to obtain services or goods. Crowdsourcing can be divided into four different strategies [49]: Crowdfunding, use of the crowd's collective financial resources; Crowdcreation, use the insight and ability of the crowd to create new products; Crowdvoting, gathering, and use of the crowd's votes regarding a certain idea or product; and, more notably, the strategy Crowd Wisdom, popularized as Wisdom of the Crowd and used in this project for the development of the task provider service. Wisdom of the Crowd states that groups of people collectively are smarter than an individual expert in problem-solving, decision-making and predicting.

An example of the direct implementation of the strategy Crowd Wisdom is the *Iowa Electronic Markets* (IEM) [12], developed by the University of Iowa and utilized by students for research and teaching purposes. IEM is a forecasting tool that allows students to invest in the outcome of future events. Due to the monetary rewards, IEM is also an incentive mechanism; making students more willing to learn and helping them make more accurate and profitable forecasts.

Another important concept closely related to Crowdsourcing is Micro Tasks or Micro Work [48], consisting of spitting larger and more complex tasks into smaller tasks that can be performed within a matter of seconds or minutes. Crowdsourcing is then used to process these tasks so that a large group of users can perform them at the same time.

When using Micro Tasks in a Crowdsourcing application, the application benefits from the user performing simple tasks or actions, while the user is usually rewarded by doing so. With the goal of this project being the development of a task solving service, this service can be characterized as a Crowd Wisdom Crowdsourcing application, while also integrating some Micro Tasking concepts and emphasizing the user interactivity aspect of it.

Crowdsourcing applications are becoming increasingly popular and widely available in a rapid-expansion market. Users are attracted to these applications in the prospect of acquiring fast and easy income.

Currently one of the most popular Crowdsourcing applications on the market is *Swagbucks* [9] where the users can earn points by watching videos or answering surveys, as shown in figure 2.1; the points can then be exchanged by *Amazon* [10] or *Paypal* [11] gift cards.

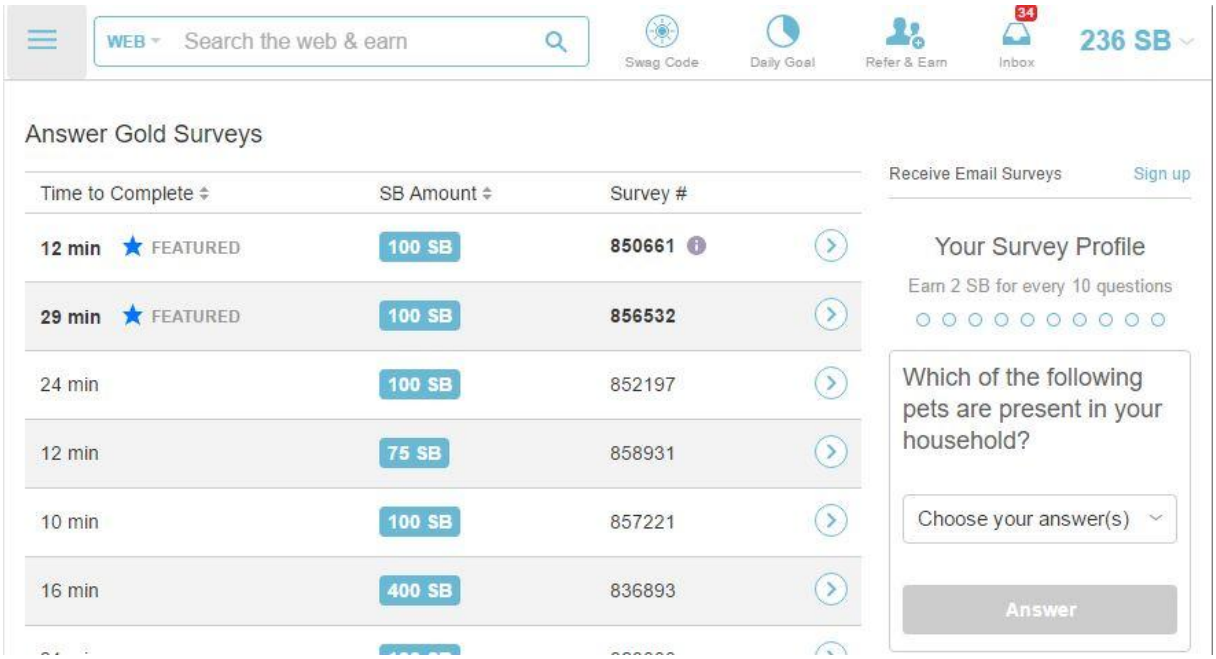


Figure 2.1: Example of surveys in *SwagBucks* [55].

Another example of a Micro Tasks Crowdsourcing application is *Mechanical Turk* [13], *Amazon*'s take on the Crowdsourcing market. Mechanical Turk serves as an intermediary between requesters that have tasks they need to be completed and workers who want to earn money by performing tasks. The task availability depends on the requesters and can be quite diverse, ranging from transcriptions to surveys, commenting on images or writing product reviews, etc. Each task to be completed is referred to as a HIT and the worker will only be paid after the HITs are approved by the requester, as observed in figure 2.2; this approval may take a few days, slowing down the payment process.

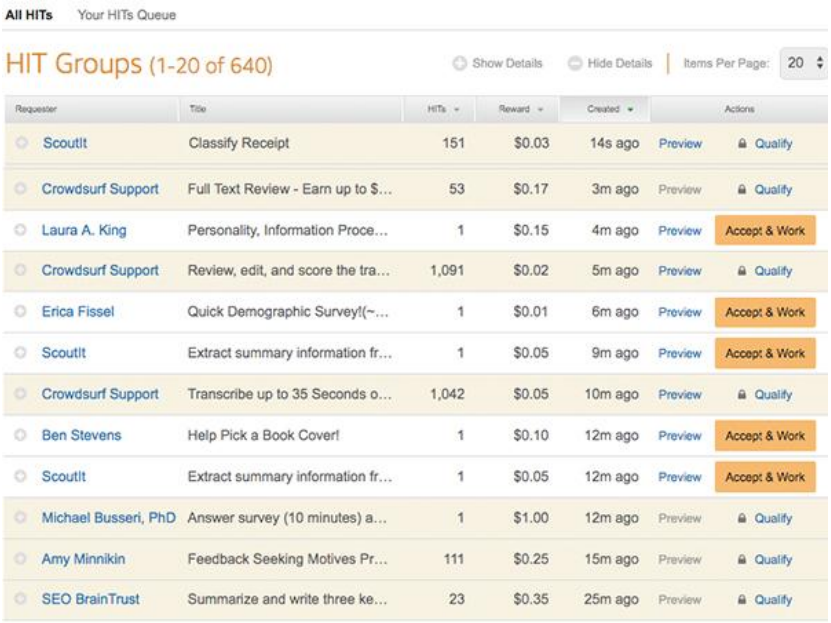


Figure 2.2: Example of hits in *Mechanical Turk* [56].

2.2 Chatbots

Most Crowdsourcing applications utilize bots to interact with the users. Bots or Internet Robots [50] are software applications that run automated tasks (scripts) over the internet. Bots normally perform simple and repetitive tasks at a higher rate than humans.

Chatbots [51] communicate with the users using auditory or textual messages, via Instant Messaging (IM), Internet Relay Chat (IRC) or another web interface, such as Facebook Bots or Twitter Bots. Chatbots may handle many tasks such as reporting the weather, sports scores or converting currency and may be used by companies to increase online engagement and streamline communication.

Chatbots offer different forms of interaction with the user [52] that may take different complexity levels. Contextual Chatbots have a higher complexity level and, using Machine Learning (ML) and Artificial Intelligence (AI), are able to self-improve by remembering conversations with specific users and grow over time. Keyword Recognition-Based Chatbots offer medium interaction complexity and use customizable keywords and AI to listen to the user, searching for specific keywords, and responding accordingly. Menu or Button-Based Chatbots, used in this project as a way for the task provider service to supply tasks, are less complex and use decision tree hierarchies for the interaction, presented to the user in the form of menus and buttons.

However, these interactions may be limited and dull due to the lack of a Chatbot personality (in lower levels of interaction complexity) or one-dimensional due to the Chatbot having only one personality (in higher levels of interaction complexity). This project attempts to address this issue with the creation of a task provider Chatbot with different personality traits for the user to choose from.

3 User Research and Design

This chapter describes the solution for this project in terms of design, as well as the user research performed and how it influenced some important design decisions made on the project.

3.1 Solution

The solution developed is named WOK – Wisdom of the Krowd, a Crowdsourcing application where users perform tasks and get rewarded by doing so. The base concept of WOK is Wisdom Of the Crowd [7][8], hence the name WOK (with *K* instead of *C*). If we apply this concept in the context of WOK, the mode of the users' answers for a certain task will most likely be the correct answer.

Tasks are supplied to the users by a task provider bot (Menu and Button-based Chatbot) that takes the form of Co-Workers of the user.

There are two types of tasks the users can perform on WOK, categorization tasks, and quality estimation tasks, see figure 3.1. In the categorization tasks the user has to assign a category and sub-category to a short text, the answer will then be a sub-category and its corresponding category. In the case of the quality estimation tasks the user will have to determine the correctness of a short text, the answer will be a Boolean yes or no answer. The task assigned to the user depends on the availability of the tasks in the system.

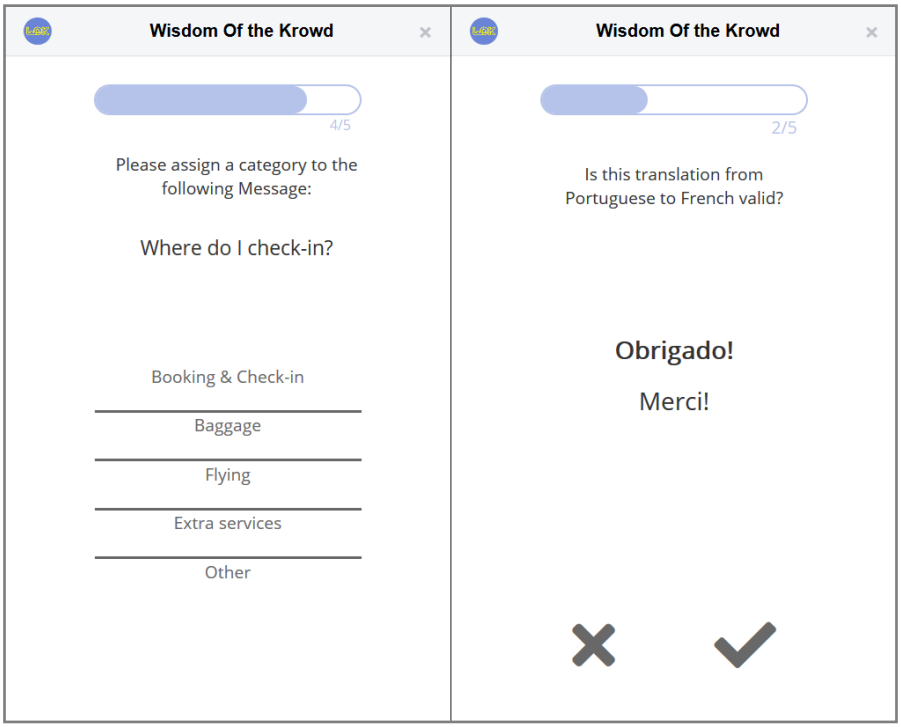


Figure 3.1: Examples of a categorization task (left) and a quality estimation task (right).

3.2 Branding

The solution has the name WOK, Wisdom of the Krowd, however, WOK is also the name of a frying pan utilized in Asian cuisine to cook noodles and the inspiration for the logo of our solution. Based on this homonym, we created a correlation between tasks being solved and noodles being cooked. The color yellow of the font derives from saffron, a yellow spice associated with Asian cuisine and the font resembles noodles, as observable from figure 3.2.

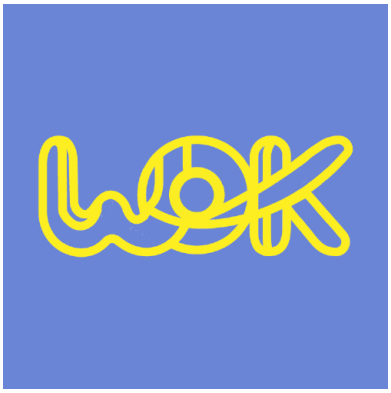


Figure 3.2: Logo of WOK.

When noodles cook they untangle, the same can be said about the tasks bring solved by the user, so, from this analogy, the slogan of WOK is “untangle”, as shown in figure 3.3.

Figure 3.3: Banner with the slogan of WOK.

3.3 User Research and Co-Workers

An innovation of WOK is the introduction of Co-Workers with different personality traits that provide tasks to users.

User research was conducted to identify the target demographic of WOK's user base. From this research, it was estimated that a demographic of adults from ages 18 to 35 composed the majority of the user base of Crowdsourcing applications; and it was assumed that WOK's users will belong to this demographic. Having the users' age group defined, more user research was performed that helped define personas to represent the users of WOK. From these user personas, Co-Workers were then created.

The task provider *bot* offers five Co-Workers for the user to choose from. It was created a persona for each of these five Co-Workers. Each of these personas has a name, a short description, and a predominant personality trait. These personality traits, in order to match and complement the users' personas, are based on the Big Five Personality Theory or Five-Factor Model[17][18], a model of an individual's personality that divides it into five traits: Openness, Agreeableness, Conscientiousness, Extraversion, and Neuroticism. So, each of the five Co-Workers has a corresponding predominant personality trait from the Big Five Personality Theory.

The available Co-Workers are the following:

Gal – Openness

Gal is a genderless bot. Gal likes long talks about the universe and the meaning of life. Gal is also interested in social and cultural matters, so tends to talk about living in communities a lot.

Figure 3.4: Avatar of Gal.

Julian – Agreeableness

Julian is a very caring bot. He likes to pep talk to his Co-Workers so they remain motivated. He also likes to chat about bot feelings and how they're perceived by humans.

Figure 3.5: Avatar of Julian.

Eli – Conscientiousness

Eli is a very formal bot. He's very work driven and is always focused on the final goal. He expects and inspires motivation through his Co-Workers, so you can call him a real role model.

Figure 3.6: Avatar of Eli.

Jesse – Extraversion

Jesse is a very outgoing bot. She likes to talk about adventures and the relationship between bots and real people. She likes to make people laugh, so she tells very funny jokes.

Figure 3.7: Avatar of Jesse.

Sam – Neuroticism

Sam is a quite shy bot. She likes to solve mysteries because it makes her feel in control. She enjoys talking about art and its repercussions in digital and real life.

Figure 3.8: Avatar of Sam.

The Co-Workers' descriptions refer to them as *bots*, as well as the Co-Workers themselves in their speech; this is not entirely true due to the existence of only one bot that manages all the five Co-Workers. This is done in order to remind the users that they are not talking to real people. If the user believes they are talking to a real person to later find out it's a bot, their trust in our product would decrease. For this reason, we wanted to explicitly inform the users that they're not talking to a real person, either by the Co-Workers explicitly referring to themselves as bots or by creating Co-Worker avatars that would not cause this confusion. This is accomplished by the use of geometric shapes and color contrast to represent the persona of the Co-Worker instead of human faces, as observable from figures 3.4 to 3.8.

Only one Co-Worker may be active at a time with Gal being the default Co-Worker, due to being the only genderless Co-Worker out of the five.

The goal of the Co-Workers is to incentivize the user to perform more tasks, thus increasing user productivity and helping the user create a bond with WOK; this can be accomplished by interacting with the user via text where each Co-Worker has a different speech dependent of its predominant personality trait.

4 Implementation

This chapter describes the developed solution, WOK, in terms of architecture, frameworks, and technologies, containing my personal contributions to the solution and explanation of the features implemented.

4.1 Architecture

WOK follows a microservices architecture [19], as shown in the diagram of figure 4.1, with three services: Facebook Handler, Server, and Client-side Handler.

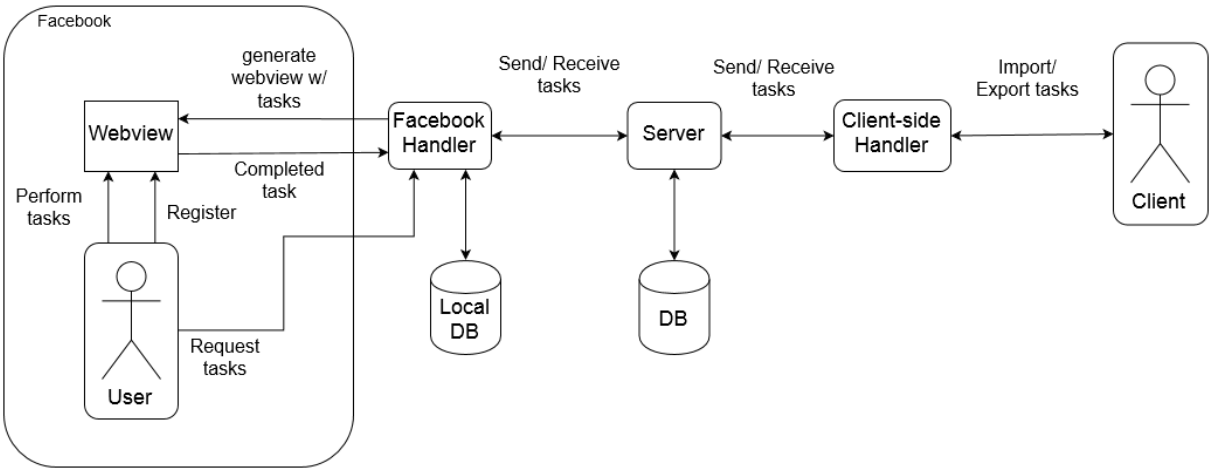


Figure 4.1: Diagram representing the architecture of WOK.

The Facebook Handler service manages the interaction with the users on the *Facebook Messenger* platform [20]. This interaction is performed by the task provider bot in the *Messenger* platform itself or by custom web pages inside the *Messenger* platform with the usage of *Webviews* [21], where the tasks are presented to the users. This service accesses a local database where *Facebook* users' information is stored.

The Server service acts as the central point of the architecture, managing the requests from the user-interface (Facebook Handler) and the client-interface (Client-side Handler). This service accesses a local database containing all the information regarding the tasks, more notably, the tasks' descriptions, completion status, users' responses and response times.

The Client-side Handler service generates a website where the clients submit new tasks to WOK and export the completed tasks, while also having access to the information of each task. The tasks can be filtered and exported to JSON [22], TSV [23] and CSV [23] file formats.

4.2 Technologies and Frameworks

Regarding the technologies and frameworks, when searching for platforms to integrate our solution WOK, platforms such as *Slack* [57] or *Line* [58] were taken into consideration due to their popularity and friendly UI, despite that, the *Facebook* platform was chosen for possessing those same features as well as providing APIs and features that improve the overall user experience.

The user interaction is performed via *Facebook* [14] and the *Messenger* [15] platform with the usage of a task provider bot while resorting to several frameworks and APIs made available by *Facebook for Developers* [16].

The task provider bot uses the Messaging framework [24] by *Facebook for Developers* and was implemented in the Python [25] development environment, as requested by *Unbabel*. The Python development environment was also used to implement the Facebook Handler, Server, and Client-side Handler services, all hosted in the Heroku [32] platform. The Flask module framework [26] and the Flask-RESTful API [27] were used for communication among the services due to their compatibility with Python.

Both the Facebook Handler local database (where the user information is stored) and the Server database (where the tasks information is stored) consist of documental databases represented using the JSON [22] format and hosted in the MongoDB mLab [33] platform. Documental databases were chosen for the representation of the users and tasks data due to the simplicity of the data and the flexibility of storage by using the same document-model format in the application code.

The tasks are presented to the human user in a *Webview*, therefore the task provider *bot* uses the *Webview* framework [21] by *Facebook for Developers*. The *Webview* itself is a browser engine embed inside Facebook that presents HTML [27] pages to the users. Several APIs by *Facebook for Developers* were also utilized to improve the user experience, such as the Persona API [29], Send API [30] and Messenger Profile API [31].

4.3 Front-end Development

My responsibility in the project was the user and client interaction and front-end development. My role in the project was closely related to design, serving as the link between the designer vision and the programmatic implementation of that vision.

In the following points, regarding the user-interface and interaction, I will explain the business model and user journey of WOK, followed by the frameworks and APIs utilized and features implemented, shown in the diagram of figure 4.2. For the client-interface and interaction, I will explain the client's sending tasks and retrieval of tasks processes in WOK, along with a detailed explanation of this module's implementation. On these points, I will also emphasize some important decisions made along the project.

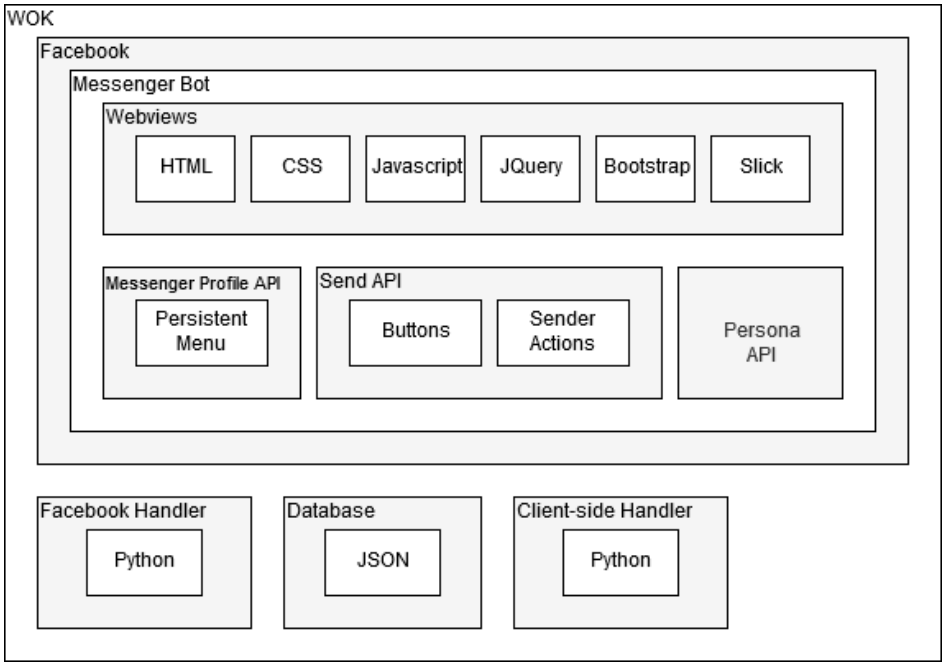


Figure 4.2: Diagram emphasizing the technologies used in the user-interface of WOK.

4.3.1 User-Interface and Facebook Interaction

The business model of the user-interface of WOK is rather simple, with the user communicating with the Co-Worker (task provider *bot*) inside the *Facebook* platform [14] via *Facebook Chat* or the *Messenger App* [15]. The task provider *bot* then communicates with the Facebook Handler Service, serving as an intermediary between the service and the user, as observed in figure 4.3.

The communication with the Co-Worker is performed via buttons. The Co-Worker's reply to the user with a text message while also providing buttons (see sub-chapter 4.3.1.5) the user can press, triggering another text response by the Co-Worker that may contain subsequent buttons. The Co-Worker's reply may contain buttons that don't trigger a response but instead open *Webviews* (see sub-chapter 4.3.1.3). *Webviews* [21] load web pages inside the *Facebook Chat* or *Messenger App* where the user can register to WOK, perform tasks and access the settings. Each action is performed in its corresponding web page inside a *Webview*.

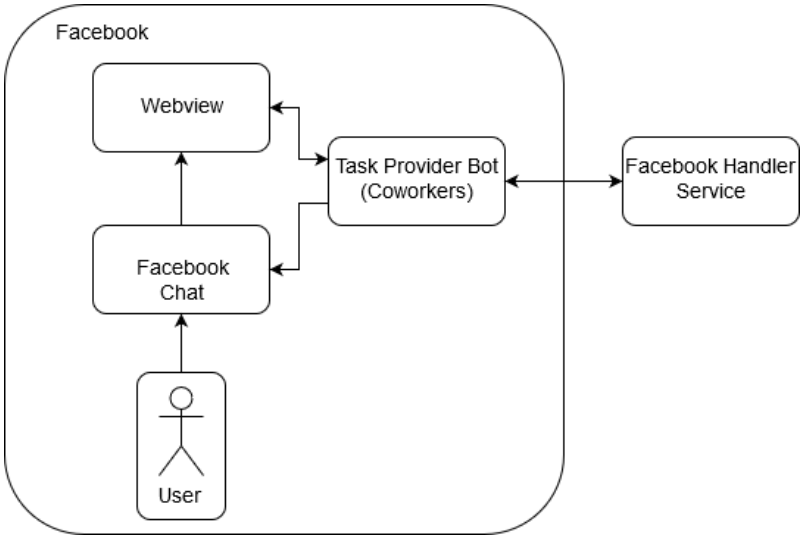


Figure 4.3: Diagram of the user-interface model of WOK.

4.3.1.1 User Journey

Concerning the user journey in WOK, to start using our application the user must first open WOK's *Facebook* page where the text chat can be accessed after accepting WOK's terms of service in the begin page, as shown in figure 4.4.

In the text chat, the user will be requested to register to WOK via a button that will open a *Webview* containing the register page where the user must provide its phone number that will be stored as an identifier of the user in our application.

Figure 4.4: Begin page.

With the registration process complete, the Co-Worker will present the user with the buttons to request tasks or open the *Webview* with the settings web page. In the settings page, the user can change his active Co-Worker, choose a nickname or enable/ disable notifications.

By pressing the request tasks button, the answer of the Co-Worker will vary, as observed in figure 4.5. If the user has a pending task, the Co-Worker will reply with a button that opens a *Webview* with a web page of that task. If the user does not have pending tasks, depending on the availability of tasks, the Co-Worker will reply with a button that opens a *Webview* containing the web page with a task of a new group of tasks for the user to complete. In the case where there are no available tasks, the Co-Worker will inform the user of this unavailability and inform the user to try again later.

Once a group of tasks is completed, the user will be presented with the tasks completion web page (containing an animation) and the Co-Worker will reply congratulating and informing the user of the coins received by the completion of the group of tasks.

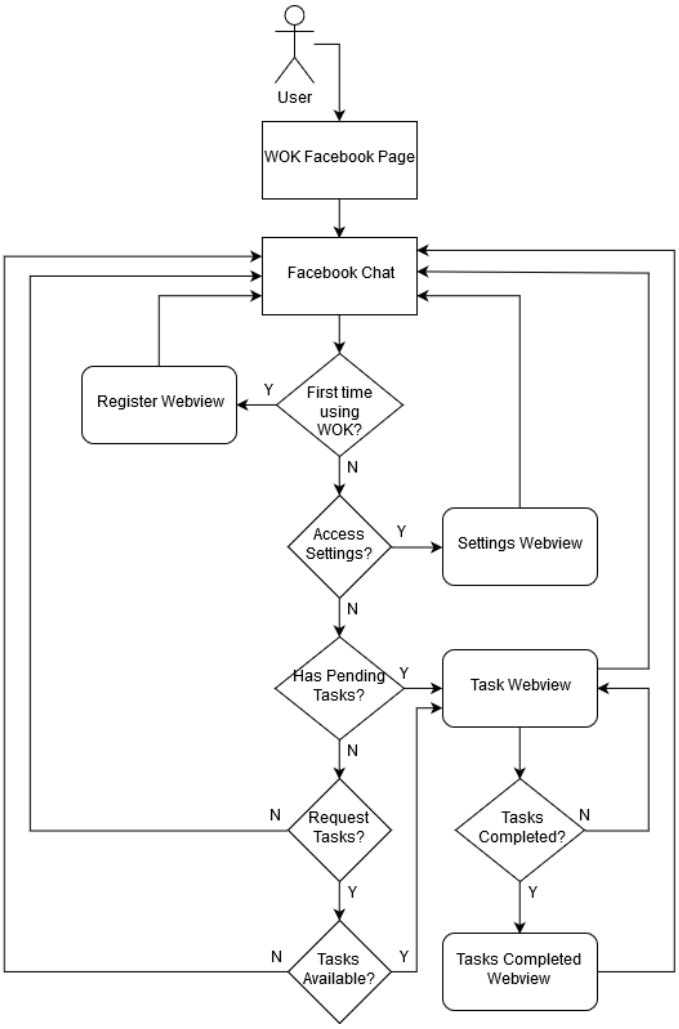


Figure 4.5: Workflow of the user journey on WOK.

4.3.1.2 Messenger Profile API – Persistent Menu

The *Messenger Profile API* [31] is used to set properties that define several aspects of the *Messenger Platform* [14][15] and features, with one of these features being the Persistent Menu [34], used in WOK.

One of the challenges of using a task provider *bot* in the *Facebook Messenger* platform was the fact that the task provider bot can't explicitly initiate a conversation with the user. The user must first interact with the task provider bot for it to respond. So a scenario where the user overloads the bot by spamming messages would be a possibility. Therefore we felt a need to control the interaction between the user and the bot. A solution found was the usage of a Persistent Menu.

A Persistent Menu is a pop-up menu with buttons through which the user interacts with the task provider *bot*, as observable in figure 4.6. The most important feature the Persistent Menu provides is the option to disable the Message composer, with the property *composer_input_disabled*, forcing the user to interact with the bot using only buttons (either the ones on the Persistent Menu or the ones sent by the Co-Worker).

Figure 4.6: Persistent menu with the button used to request tasks.

Asides from the *composer_input_disabled* property, another relevant property worth mentioning is the *call_to_actions* property, consisting of an array of top-level menu items for the persistent menu. In the context of WOK, this array will only contain one item, the “Request Tasks” action.

4.3.1.3 Webview and Web Pages

Webviews [21] are a feature of the Messenger Platform [14][15] where web pages can be loaded inside the Messenger.

A challenge faced was how to represent tasks and where the user would perform said tasks. Since it would be difficult to represent tasks with message bubbles or buttons in the text chat, *Webviews* were chosen to help with this representation.

Webviews consist of embed browser engines where web pages are loaded inside the *Facebook Messenger Platform*. The tasks themselves are presented to the user in HTML pages inside its *Webview*, as well as the register page, settings page, and the tasks completion page, allowing more flexibility and freedom in the tasks' representation.

The web pages available to the user are the following:

Categorization Task – where the user performs categorization tasks. It contains the task description, the list of categories, a list of sub-categories for each corresponding category and a progression bar to track the group of tasks progression, see figure 4.7.

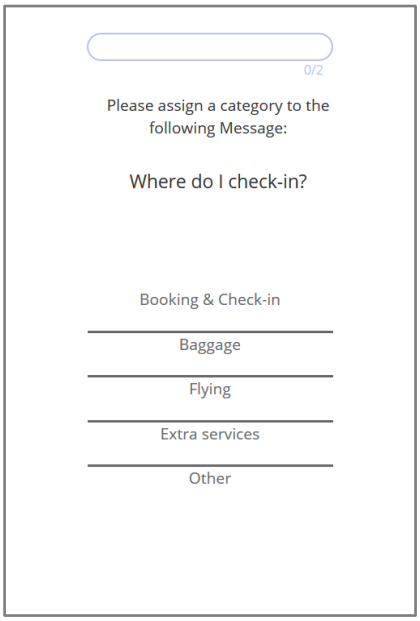


Figure 4.7: *Webview* with the web page of a categorization task.

Quality Estimation Task – where the user performs quality estimation tasks. It contains the task description, the short text whose correctness needs to be determined, two buttons to select the Boolean answer and also the progression bar, see figure 4.8.

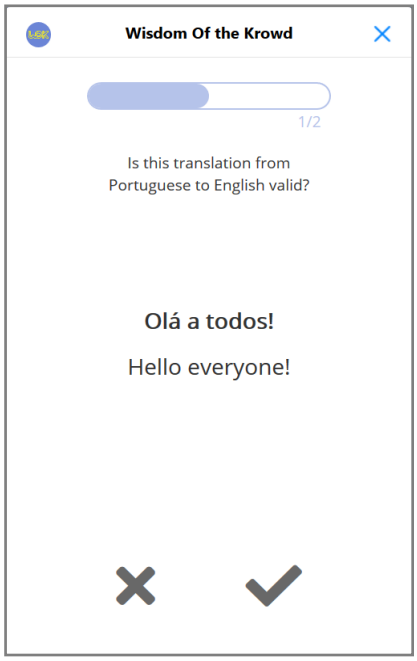


Figure 4.8: *Webview* with the web page of a quality estimation task.

Tasks Completion – is a web page that congratulates the user on completing a group of tasks with a *GIF* [35] file, as shown in figure 18. The *Webview* is closed after the GIF completes the animation, see figure 4.9.

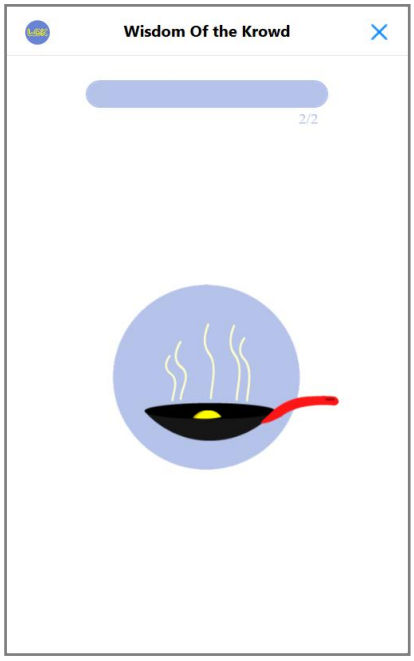


Figure 4.9: *Webview* with the tasks completion web page.

Settings – where the user can change the active Co-Worker, as well as the nickname and notification options. It contains a dynamic carousel for the user to select the active Co-Worker, a text box where the user can change his nickname and a toggle to enable/disable notifications, see figure 4.10.

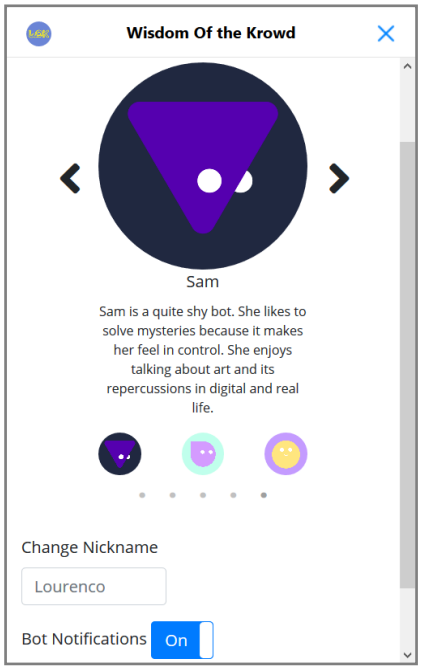


Figure 4.10: *Webview* with the settings web page.

Registration – where the user must first register to start WOK. It contains text boxes the user fills with his nickname and phone number (to be used as an identifier), see figure 4.11.

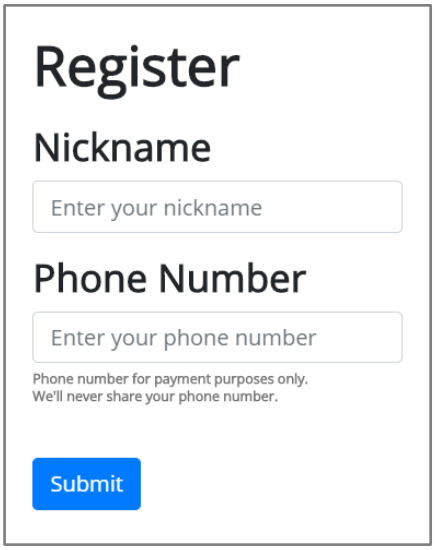


Figure 4.11: *Webview* with the register web page.

The registration process is only performed by the user once, so the Registration web page is only accessed once, during this process. Once the registration process is complete, all other pages become accessible to the user, as observable in the navigation diagram of figure 4.12. The user accesses the Task Completion web page only when the last task of a group is completed, so this web page is dependent on the Categorization Task page or the Quality Estimation Task page (depending on the type of the last task of the group). The Settings web page is independent and can be accessed at any time.

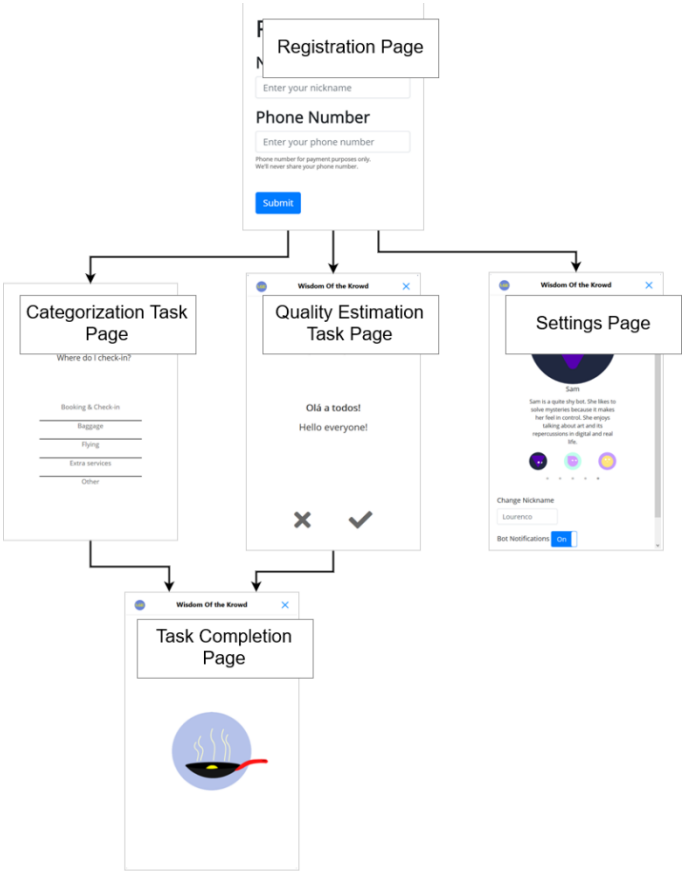


Figure 4.12: General navigation diagram.

The web pages where the tasks are presented to the user, as well as the register, settings, and task completion pages don't consist of only plain HTML [28], other libraries, plug-ins, and frameworks were used. Regarding the Categorization and Quality Estimation tasks web pages particularly, the generation of these web pages is dynamic using two templates, one for each task type. The templating engine Jinja2 [36] was used, allowing the writing of code with a syntax similar to Python [25] with the usage of special placeholders in the template.

CSS [37] was used to change and improve the visual aspect of the web pages. The JavaScript [38] development environment was used on all web pages to obtain the *recipient_id* of the user (so that the Facebook Handler Service identifies the user) and also to implement many other features.

JQuery [39], a JavaScript library, was also used on all web pages for obtaining the *recipient_id*. It was also used in the quality estimation tasks web page to cause the text presented to the user to fade in; in the categorization tasks web page for the lists of categories and sub-categories animation; and on the tasks completion web page, as well as both the categorization and quality estimation tasks web pages, to create a progression bar that tracks the group of tasks progression, observable in figure 4.13.

Figure 4.13: Progression bar.

The framework Bootstrap [40] was used on all the web pages to improve the overall visual aspect, making the web pages more appealing to the users.

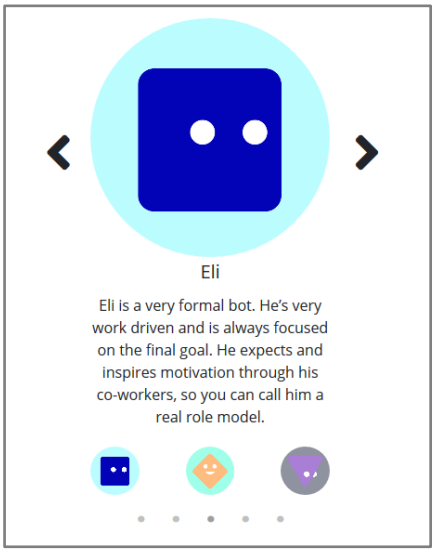


Figure 4.14: Carousel for Co-Worker selection.

Slick [41], a Javascript plug-in, was used to create a dynamic carousel where the user can select a Co-Worker, as shown in figure 4.14. Some changes were made to the Javascript code of the base carousel to obtain the desired aspect, specifically the arrows to switch between the Co-Workers and the bottom bar where all their avatars are displayed.

4.3.1.4 Persona API

The Persona API [29] allows the creation of Personas [42] in a bot that carries messaging conversations with end-users in the *Messenger Platform* [14][15]. It is still the bot that sends the message in the platform API, but by using this API, there is the option to send the message under the name of a Persona instead of the bot.

With the usage of this API, it was possible to very easily integrate each of the Co-Workers as Personas in the *Messenger Platform*.

For each Persona created, a new icon will be shown with the Co-Worker avatar and all messages sent by the Persona will be accompanied by an annotation above the message that states the name of the Persona and the name of the page the Persona belongs to (in this case WOK, Wisdom of the Krowd), as shown in figure 4.15.

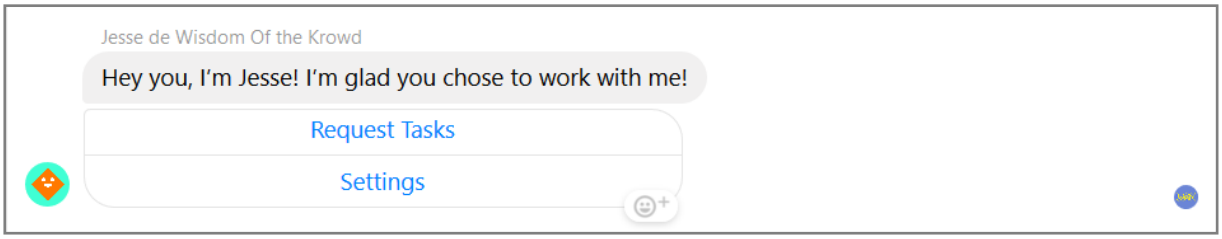


Figure 4.15: Co-Worker Jesse as a Persona in the Messenger Platform.

The API allows the creation of Personas by specifying the following properties on the request: *name*, corresponding to the name of the persona and, *profile_picture_url*, a URL of the avatar icon associated to the Persona. The response will contain the property *id*, the unique identifier of the Persona.

The Persona API also allows the retrieval or deletion of a specific persona by passing the unique identifier of the Persona in the request URL.

To send a message with the Persona, the following properties need to be specified on the request: *recipient*, containing the identifier of the user on the Facebook chat; *message*, containing the message to be sent; and *persona_id*, the unique identifier of the Persona.

4.3.1.5 Send API – Sender actions and buttons

The Send API [30] is the main API used by WOK to send text messages to users along with buttons and sender actions.

Sender actions [43] are used to simulate human interaction either by marking messages sent by the user as seen or by simulating the Co-Worker is typing a message instead of replying instantly, as shown in figure 4.16, thus making the task provider bot feel more responsive. Along with the usage of sender actions, a delay time is defined to simulate the Co-Worker typing. This delay is variable, depending on the message complexity, with smaller messages having a smaller delay duration of 0,5 seconds, while lager and more complex messages have a larger delay duration of 2,5 seconds.

Sender actions are specified as the property *sender_action* inside the message payload, taking the following values: *mark_seen*, marking the last message sent by the user as read; *typing_on*, turning on the typing indicator, shown in figure 24; and *typing_off*, turning off the typing indicator.

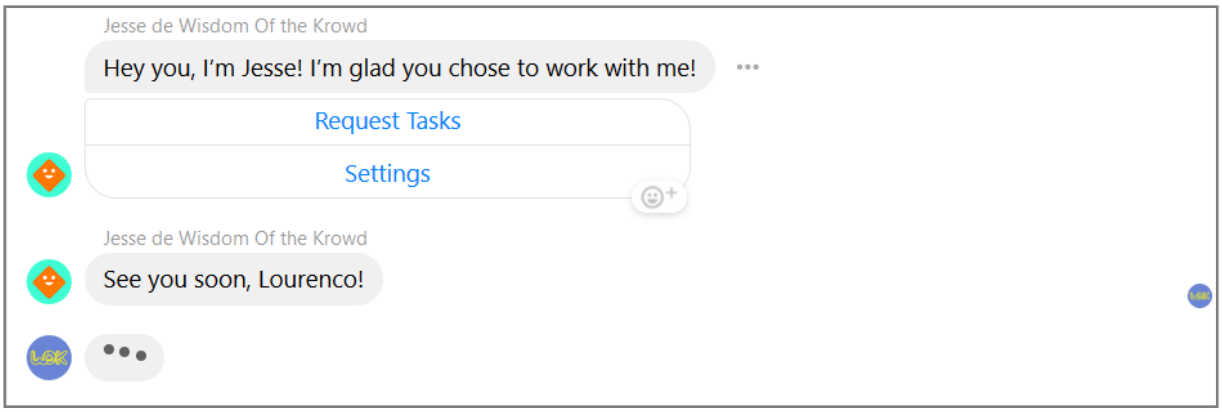


Figure 4.16: Sender action simulating the Co-Worker typing a message.

The Send API is also responsible for the buttons [44] sent in the Co-Workers' replies. URL buttons, a type of button that open URLs on a Webview, were the most commonly used. These buttons are sent inside the message payload as an attachment, consisting of an array of button objects characterized by the following properties: *type*, the type of the button, in this case, *web_url*; *title*, the text to present in the button; and *url*, representing the URL of the web page to open in a Webview.

4.3.2 Client-interface and Interaction

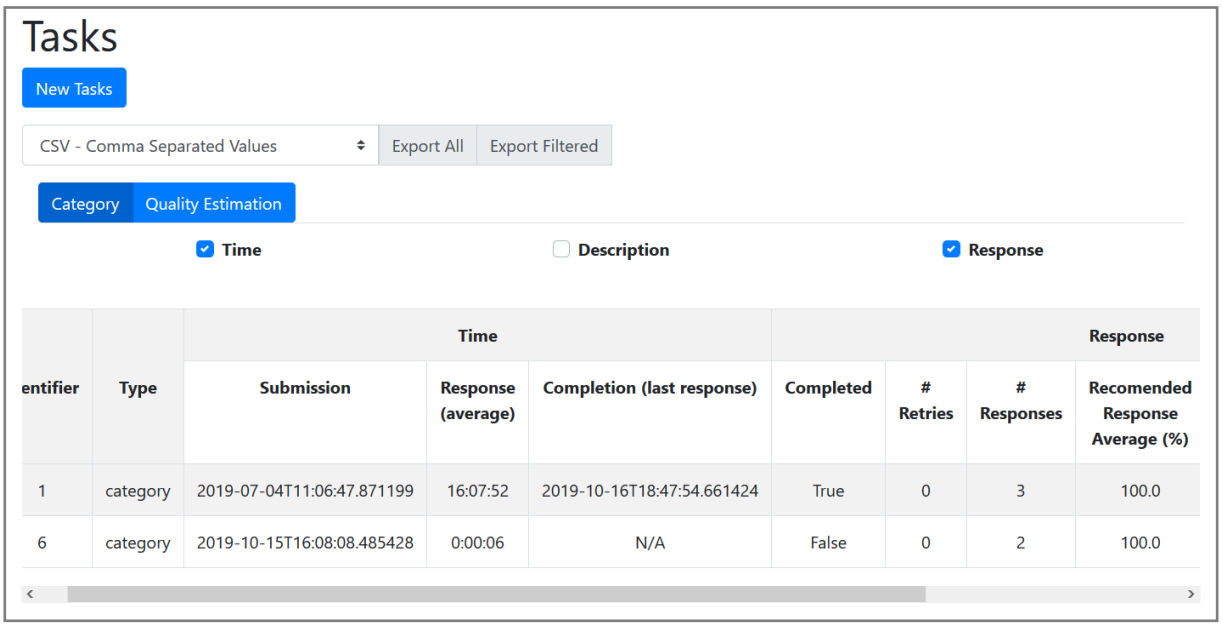
The Client-side handler service is responsible for the client-interface and interaction. Using this service the clients can submit new tasks to be performed by the users on the *Facebook* platform [14][15] and obtain the results of the completed tasks.

The interface for the task submission and retrieval consists of web pages and the Client-side handler service is a server that manages these web pages. There are two web pages: the task retrieval web page and the task submission web page, consisting of HTML [28] pages that use JavaScript [38], CSS [37], Bootstrap [40] and JQuery [39].

4.3.2.1 Task Retrieval

The task retrieval page allows the user to visualize and filter task data by three different categories: time, description and response, as shown in figure 4.17. The tasks and these categories are displayed in the form of tables generated from a template, using the templating engine Jinja2 [36]. The task time category shows information of the task date and time of submission, average response time and date and time of completion. The task description category displays the task description and word count. And lastly, the task response category showcases information regarding the users' answers and completion such as the number of responses for the task, if the task is completed, the preferred answer for the task and the recommended response average.

The recommended response average percentage is an important measurement of the quality of the response, the lower the value, the more scattered the responses are and the less accurate the preferred answer may be.



entifier	Type	Time			Completed	# Retries	# Responses	Response
		Submission	Response (average)	Completion (last response)				Recomended Response Average (%)
1	category	2019-07-04T11:06:47.871199	16:07:52	2019-10-16T18:47:54.661424	True	0	3	100.0
6	category	2019-10-15T16:08:08.485428	0:00:06	N/A	False	0	2	100.0

Figure 4.17: Task retrieval web page.

Some task data varies depending on the type of the task, for quality estimation tasks, the response category will contain a true or false Boolean as the answer and, in the case of categorization tasks, the most chosen category and sub-category will be the answer. For this reason, each type of task has its table where the tasks of that type are presented and the client can alternate between the tables.

In the task retrieval page, the client can obtain information of the tasks presented in the tables using the export option. By using the “Export All” and “Export Filtered” buttons the client obtains a file with the data of the currently selected task category table in the desired format. The available formats are CSV [23], TSV [23] and JSON [22], as observable in figure 4.18.

The “Export All” and “Export Filtered” actions are performed in runtime on the client browser with JavaScipt functions. The task data, presented to the client in the web page tables, is converted to the desired format for the client to download. The conversion process utilizes HTML DOM [45] to create the downloadable file, as well as the templating engine Jinja2 [36] to cycle through the tasks.

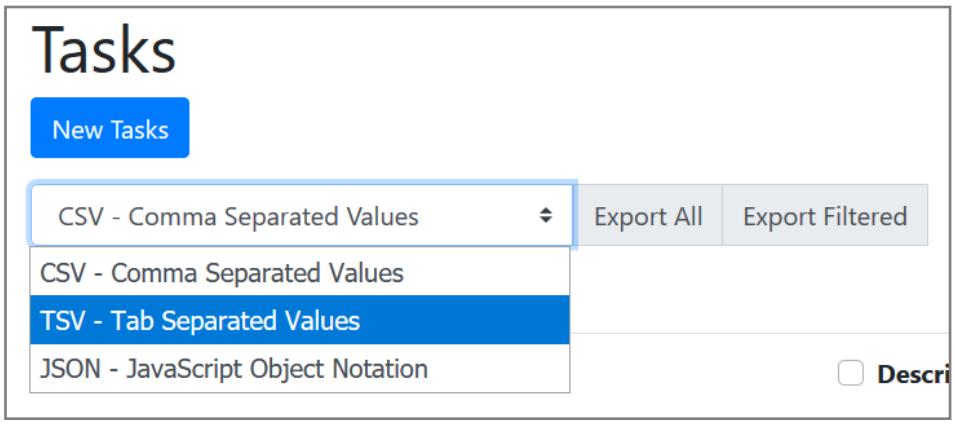


Figure 4.18: Formats of tasks available to export.

The Client-side Handler service presents the task retrieval web page to the client after the tasks data is requested to the Server service. Once the tasks data is obtained from the Server service in JSON format, it must be extracted and converted to a representation that matches the columns of the task retrieval page table because, although it maintains the JSON format, the representation differs. Once this process is complete and the tasks data matches the columns, the task retrieval web page is generated from the template and rendered.

4.3.2.2 Task Submission

The task submission page allows the client to submit new tasks to WOK by uploading a file with the tasks, as shown in figure 4.19. In this page, the client can select the type of tasks to submit: Categorization or Quality Estimation; and the format of the tasks file: JSON, CSV, and TSV.

A Quality Estimation task is characterized by the task description, a source message, and a target message. A Categorization task is characterized by a task message, a list of categories and, for each category of the list, a corresponding list of subcategories.

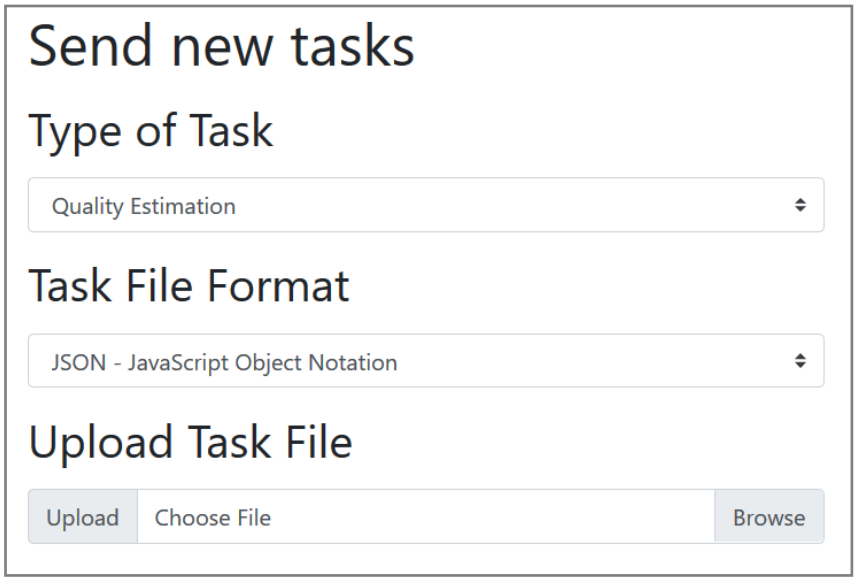


Figure 4.19: Task submission page.

It is worth noting the task submission page allows Quality Estimation tasks to be uploaded in JSON, CSV, and TSV format, but for Categorization tasks to be uploaded to WOK only the JSON format is accepted. This limitation was imposed due to the fact that the number of categories and subcategories for this type of task can be highly variable. So by using CSV and TSV formats to represent this type of tasks due to this high variance and each task being represented in one line, it would be difficult to determine the number of categories and subcategories for each task.

For the Client-side Handler service to send the Server service new tasks it must first process the file received by the client containing the tasks. If the tasks are in CSV or TSV formats, they are converted to the JSON format. If the tasks are already in the JSON format, its representation is changed to match the one of the Server service. Once the tasks are in the JSON format with the correct representation, they are sent to the Server service to be stored and sent to clients.

5 Discussion

Some important decisions were made during the development of WOK are worth discussing.

An initial idea was to present the tasks to the user in the message sent by the Co-Workers and present several buttons, provided by the Messaging framework [20], where the user could solve the task. But by using this approach, the Messaging framework had a limitation on the number of buttons a message can contain, so for some categorization tasks it would not be possible to fit all categories in a single message. A solution to this problem was the usage of *Webviews* [21].

During the first session of user testing, we concluded from the users' feedback the persistent menu was unintuitive. So, although tasks can still be requested from this menu, we shifted to an approach where the user requests tasks and accesses the settings page using buttons sent with the Co-Worker message. After the user testing session and some market research, it was also concluded the average price users are willing to receive for performing a task is around 0,10 €.

Several features were introduced to make WOK more appealing to the users, such as the Task completion page, the introduction of Co-Workers and a nickname for the user. The main goal of the Co-Workers is to improve user productivity, incentivizing the users to perform more tasks along with techniques such as the grouping of tasks and the progression bar for the user to track the remaining tasks of a group.

The project was implemented using an iterative development process with weekly iterations and weekly meetings performed at the end of each iteration. The main goals of the project were divided into small tasks to be performed during these iterations. During each weekly meeting, the tasks of the previous weekly iteration were reviewed and the tasks for the next weekly iteration were planned.

6 Conclusion

The project developed, WOK, successfully integrates the concept of Wisdom Of the Crowd [6][7] in a Crowdsourcing service integrated in the *Facebook* platform [14][15]. Users can register to WOK and perform categorization and quality estimation tasks. Clients can submit new tasks to WOK and retrieve the completed tasks.

During the development of WOK two user testing sessions were performed. However, due to the small sample size of 5 people during each session, the feedback obtained may not enough to fully determine what improvements need to be made to WOK. To fully grasp the public perception of WOK and what changes need to be performed to appeal to a larger audience, larger-scale testing sessions are needed, along with alpha and beta testing.

The next step on the development of WOK consists in implementing several features and functionalities missing in the current version. These features consist of the user payment by integrating payment platforms such as *Paypal* [11] or *MbWay* [47] with WOK so that the coins the users receive can be converted into real currency. Due to this feature not being implemented in the current version, WOK was not made available to the general public. Another feature consists of the client authentication so that the client could obtain and access the tasks he published to WOK after inserting his authentication credentials in task retrieval and submission web pages.

This project was an enriching experience both academically and professionally allowing me to experience a business environment for the first time while cooperating with colleagues from different fields of studies to form a diverse and insightful team.

References

1. Capstone Model definition, https://www.edglossary.org/capstone-project/, last accessed 2019/10/27.
2. Instituto Superior Técnico Homepage, https://tecnico.ulisboa.pt/en/, last accessed 2019/10/27.
3. Unbabel Homepage, https://unbabel.com/, last accessed 2019/10/27.
4. Schenk, Eric; Guittard, Claude: Crowdsourcing What can be Outsourced to the Crowd and Why, https://www.researchgate.net/publication/40270166_Crowdsourcing_What_can_be_Outsourced_to_the_Crowd_and_Why.
5. Estellés-Arolas, Enrique; González-Ladrón-de-Guevara, Fernando: Towards an Integrated Crowdsourcing Definition, http://www.crowdsourcing-blog.org/wp-content/uploads/2012/02/Towards-an-integrated-crowdsourcing-definition-Estell%C3%A9s-Gonz%C3%A1lez.pdf.
6. Soll, Jack; Mannes, Albert; Larrick, Richard: The "Wisdom of Crowd" Effect, https://faculty.fuqua.duke.edu/~jsoll/Soll,%20Mannes,%20Larrick%202011.pdf.
7. On the Wisdom of Crowds: Collective Predictive Analytics, https://towardsdatascience.com/on-the-wisdom-of-crowds-collective-predictive-analytics-302b7ca1c513.
8. Belas Artes Lisboa Homepage, https://www.belasartes.ulisboa.pt/, last accessed 2019/10/27.
9. Swagbucks Homepage, https://www.swagbucks.com/, last accessed 2019/10/27.
10. Amazon Homepage, https://www.amazon.com/, last accessed 2019/10/27.
11. Paypal Homepage, https://www.paypal.com, last accessed 2019/10/27.
12. Iowa Electronic Markets Homepage, https://iemweb.biz.uiowa.edu/, last accessed 2019/10/27.
13. Mechanical Turk Homepage, https://www.mturk.com/, last accessed 2019/10/27.
14. Facebook Homepage, https://www.facebook.com/, last accessed 2019/10/27.
15. Messenger Homepage, https://www.messenger.com/, last accessed 2019/10/27.
16. Facebook for Developers Homepage, https://developers.facebook.com/, last accessed 2019/10/27.
17. Rothmann S, Coetzer EP: The big five personality dimensions and job performance, https://sajip.co.za/index.php/sajip/article/view/88.
18. Poropat AE: A meta-analysis of the five-factor model of personality and academic performance, https://www.ncbi.nlm.nih.gov/pubmed/19254083.

19. Microservice Architecture, https://microservices.io/patterns/microservices.html, last accessed 2019/10/27.

20. Facebook for Developers Messenger Platform page, https://developers.facebook.com/docs/messenger-platform/, last accessed 2019/10/27.

21. Facebook for Developers Webview page, https://developers.facebook.com/docs/messenger-platform/webview/, last accessed 2019/10/27.

22. JSON Homepage, https://www.json.org/, last accessed 2019/10/27.

23. CSV and TSV formats, https://help.gnome.org/users/gnumeric/stable/gnumeric.html#file-format-csv, last accessed 2019/10/27.

24. Facebook for Developers Messaging framework page, https://developers.facebook.com/docs/messenger-platform/send-messages/, last accessed 2019/10/27.

25. Python Homepage, https://www.python.org/, last accessed 2019/10/27.

26. Flask documentation page, https://flask.palletsprojects.com/en/1.1.x/, last accessed 2019/10/27.

27. Flask-RESTful documentation page, https://flask.palletsprojects.com/en/1.1.x/, last accessed 2019/10/27.

28. HTML W3C page, https://www.w3.org/html/, last accessed 2019/10/27.

29. Facebook for Developers Persona API page, https://developers.facebook.com/docs/messenger-platform/reference/personas-api/, last accessed 2019/10/27.

30. Facebook for Developers Send API page, https://developers.facebook.com/docs/messenger-platform/reference/send-api/, last accessed 2019/10/27.

31. Facebook for Developers Messenger Profile API page, https://developers.facebook.com/docs/messenger-platform/reference/messenger-profile-api/, last accessed 2019/10/27.

32. Heroku Homepage, https://www.heroku.com/, last accessed 2019/10/27.

33. mLab Homepage, https://mlab.com/, last accessed 2019/10/27.

34. Facebook for Developers Persistent Menu page, https://developers.facebook.com/docs/messenger-platform/reference/messenger-profile-api/persistent-menu, last accessed 2019/10/27.

35. GIF W3C documentation page, https://www.w3.org/Graphics/GIF/spec-gif87.txt, last accessed 2019/10/27.

36. Jinja2 documentation page, https://jinja.palletsprojects.com/en/2.10.x/, last accessed 2019/10/27.

37. CSS W3C page, https://www.w3.org/Style/CSS/, last accessed 2019/10/27.

38. JavaScript Homepage, https://www.javascript.com/, last accessed 2019/10/27.

39. JQuery Homepage, https://jquery.com/, last accessed 2019/10/27.

40. Bootstrap Homepage, https://getbootstrap.com/, last accessed 2019/10/27.

41. Slick Homepage, https://kenwheeler.github.io/slick/, last accessed 2019/10/27.

42. Facebook for Developers Personas, https://developers.facebook.com/docs/messenger-platform/send-messages/personas/, last accessed 2019/10/27.

43. Facebook for Developers Sender Actions, https://developers.facebook.com/docs/messenger-platform/send-messages/sender-actions/, last accessed 2019/10/27.

44. Facebook for Developers Buttons, https://developers.facebook.com/docs/messenger-platform/send-messages/buttons, last accessed 2019/10/27.

45. DOM documentation, https://dom.spec.whatwg.org/, last accessed 2019/10/27.

46. Ikediego, Henry; Ikan, Mustafa; Abubakar, A. Mohammed; Bekun, Festus: Crowd-sourcing (who, why and what), https://www.emerald.com/insight/content/doi/10.1108/IJCS-07-2017-0005/full/html.

47. MBWay Homepage, https://www.mbway.pt/, last accessed 2019/10/27.

48. Microtasking definition, https://www.clickworker.com/crowdsourcing-glossary/microtasking-microjobs/, last accessed 2019/10/27.

49. Crowdsourcing strategies, https://www.tricider.com/Crowdsourcing-Strategies, last accessed 2019/10/27.

50. Bot definition, https://techterms.com/definition/bot, last accessed 2019/10/27.

51. Chatbot definition, https://searchdomino.techtarget.com/definition/IM-bot, last accessed 2019/10/27.

52. Chatbot interaction, https://chatbotsmagazine.com/the-3-types-of-chatbots-how-to-determine-the-right-one-for-your-needs-a4df8c69ec4c, last accessed 2019/10/27.

53. Howe, Jeff: Why the Power of the Crowd is Driving the Future of Business, https://dl.acm.org/citation.cfm?id=1481457.

54. Brabham, Daren: Crowdsourcing, http://wtf.tw/ref/brabham.pdf.

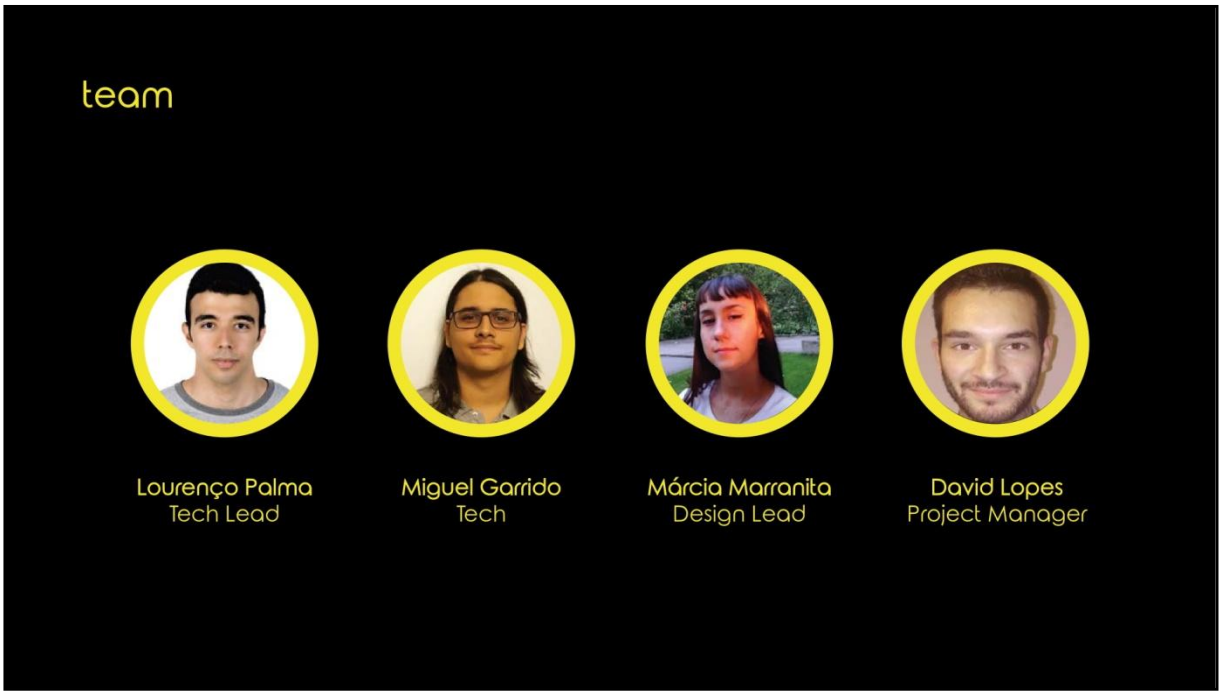
55. Example of surveys in Swagbucks, https://www.doughroller.net/reviews/swagbucks-review-earn-rewards-surf/, last accessed 2019/10/27.

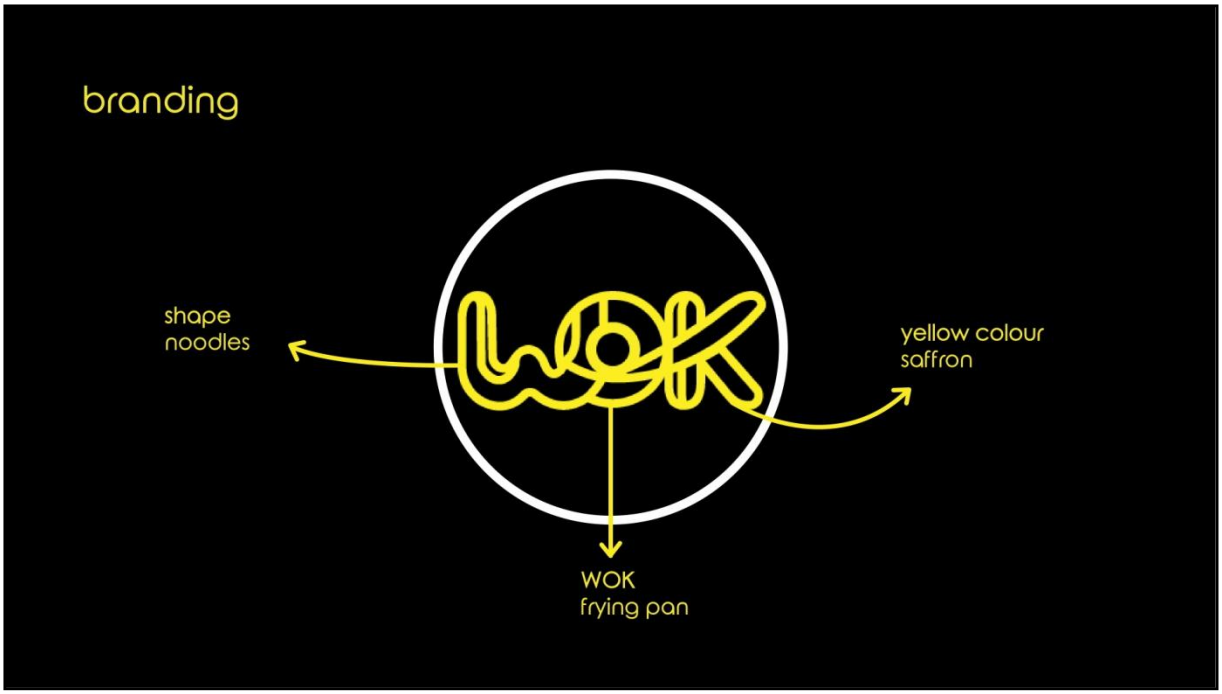
56. Example of hits in Mechanical Turk, https://thehustle.co/making-money-on-amazon-mechanical-turk/, last accessed 2019/10/27.

57. Slack Homepage, https://slack.com/, last accessed 2019/10/27.

58. Line Homepage, https://line.me/en/, last accessed 2019/10/27.

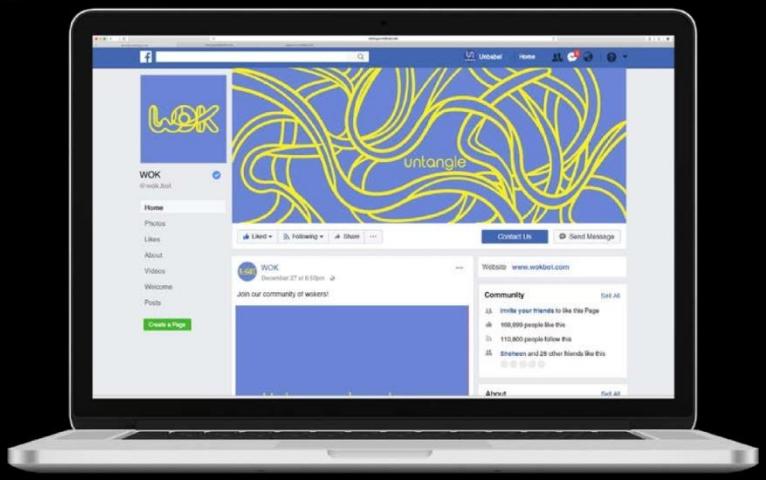
Annex A – Group Report




branding
shape
noodles
yellow colour
saffron
WOK
frying pan


untangle
untangle
(slogan)
free from a tangled or twisted state;
make (something complicated or confusing)
easier to understand or deal with.

facebook page



avatars

charatcteristics

feminine — masculine

Agreeableness — Extraversion — Openness — Neuroticism — Conscientiousness

default
gender: gender neutral
personality type: openness

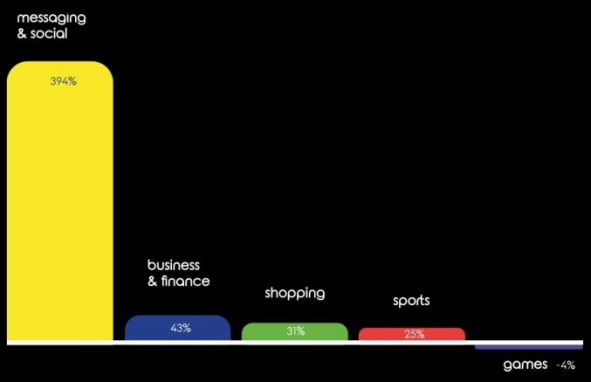
*Big Five Theory
Empirical base to develop bot's personalities.
There are five types of personalities depicted and placed in the gender spectrum above.

personality

Gal is a genderless bot.
Gal likes long talks about the universe and the meaning of life.
Gal is also interested in social and cultural matters, so tends to talk about living in communities a lot.

Jesse is a very outgoing bot. She likes to talk about adventures and the relationship between bots and real people.
She likes to make people laugh, so she tells very funny jokes.

messaging services



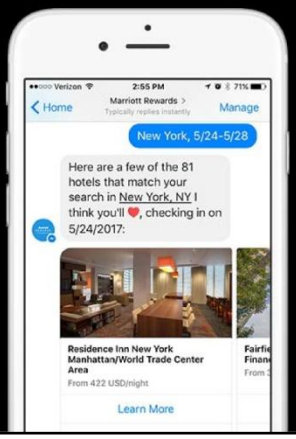
why messaging services?

1. app time spent growth 69%
2. user growth

chat bots

Bot applications are many:
1. customer support, for marketing;
2. share news;
3. financial market.

ex.: Marriott International



crowdsourcing

"the act of taking a job traditionally performed by a designated agent (usually an employee) and outsourcing it to an undefined, generally large group of people in the form of an open call"

Jeff Howe

crowdsourcing categories

service marketplaces

microtasks

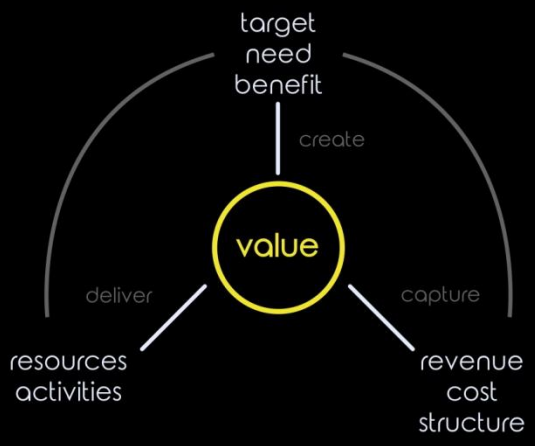
crowd design

crowdfunding

prediction markets

business model

*The way that a project or a company creates and captures value



business model

10 key parameters

1. Customer segments;
2. Value proposition;
3. Channels;
4. Customer relationships;
5. Key activities;
6. Key partners;
7. Key resources;
8. Cost Structure;
9. Revenue Streams;
10. Crowd rewards.

methods

1. Competitive analysis;
2. Business model canvas;
3. Study cases.

business model

categorization tasks

A global idea of this part of the project is using the crowd to categorize tasks. The client will send the "jobs", that are sentences that have to be categorized , and the crowd will be the workforce. After that will pass for a process to validate their responses to guarantee the quality of the categorization.

categorization tasks

price per task

CrowdFlower ~ $ 0.10

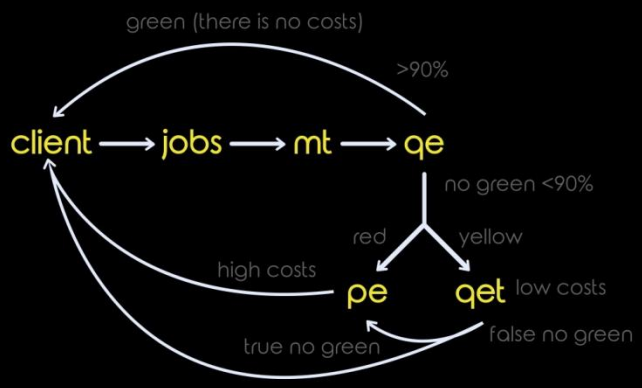
per task

defined crowd ~ $ 0.33

business model

quality estimation tasks

A global idea about what is being developed in this one: a client will send jobs to the company, in the case of unbabel will be translation jobs. This ones will be sent to a machine translation that will do the translation and the results will be evaluated by a quality estimator. In this one there is a threshold that will defined if that was a good translation, green, or if it's bad, no green.
If we got some doubtful results, that will enter in the quality estimator tasks(QET). Using crowdsourcing, we will get the results to evaluate if the results are really bad, and then go to the PE. If the results come green afterwards that will be sent to the client.

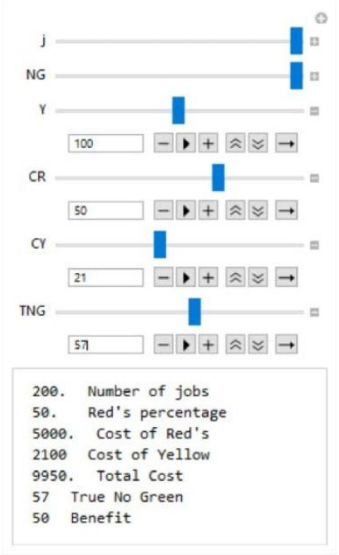


quality estimation tasks

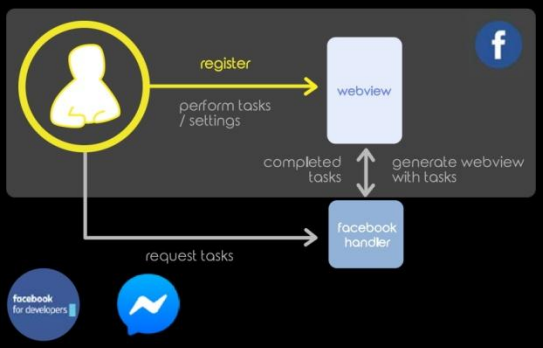
simulation

As we can see, is possible to have benefit with this project.

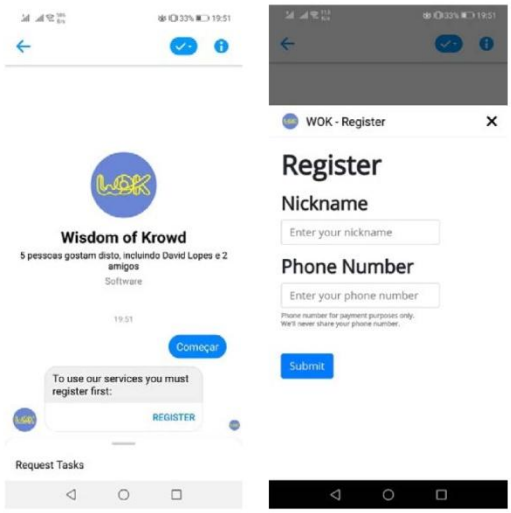
Wolfram Mathematica



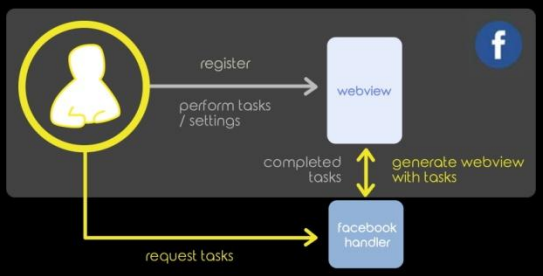
user side architecture



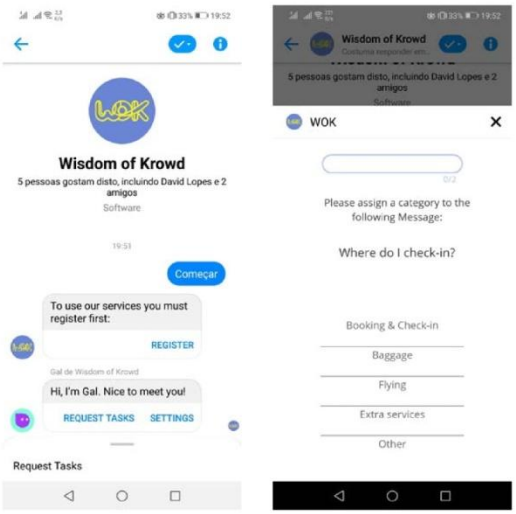
-the bot uses facebook for developers framework
-the communciation with the user is performed inside messenger platform only
-user must first register with its nickname and phone number



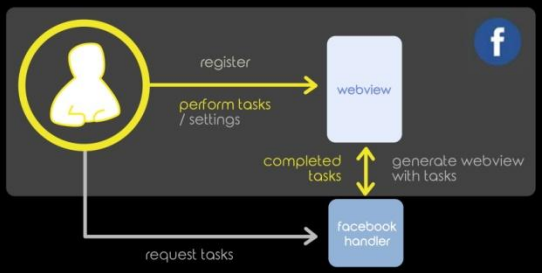
user side architecture



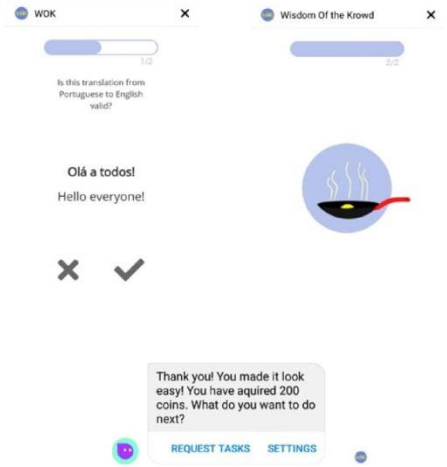
-the personas are implemented using the persona API by Facebook for developers
-the personas provide tasks to the users
-these tasks are presented to the user in webviews using the webview API
-webviews works similarly to a embed web pages inside messenger platform, working for both desktop and mobile
-the webpage itself is an html page



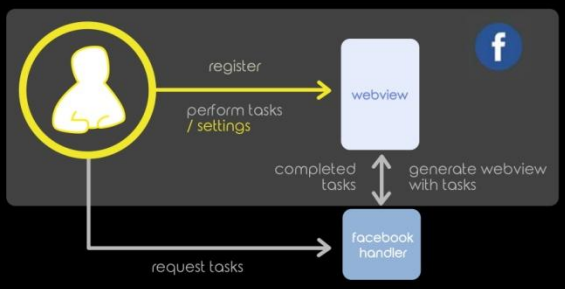
user side architecture



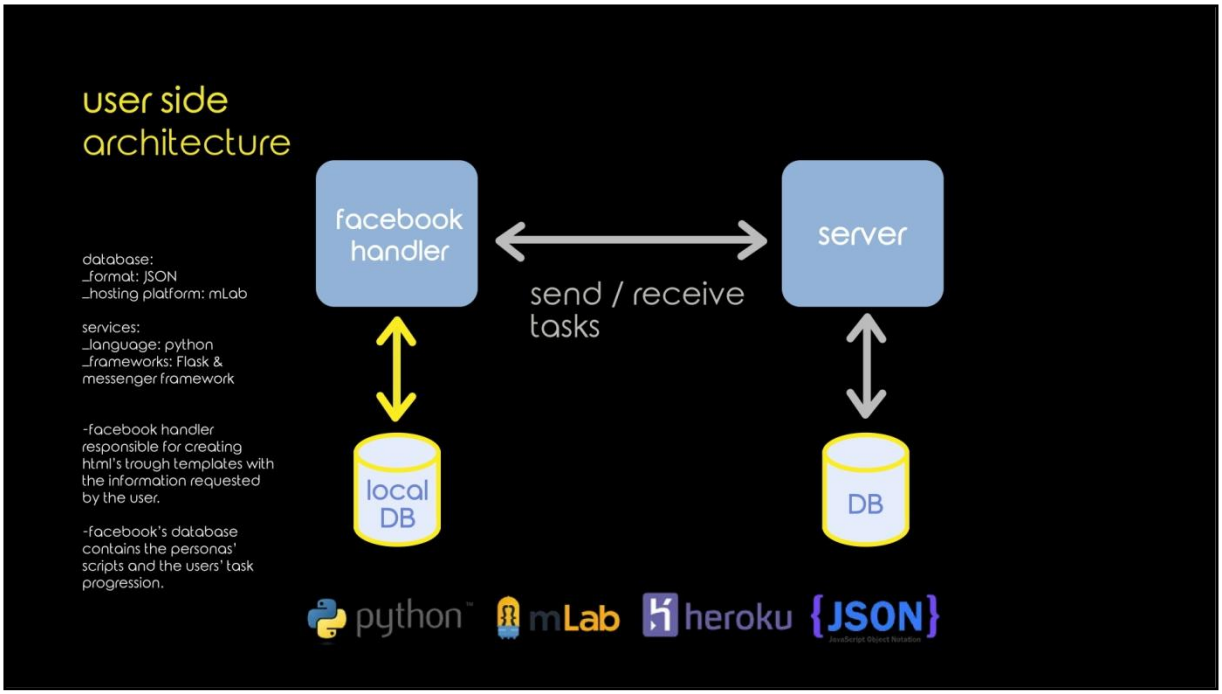
-the user must complete groups of tasks instead of singular tasks
-the tasks can be either categorization or quality estimation type
-upon completion the persona rewards the user with coins

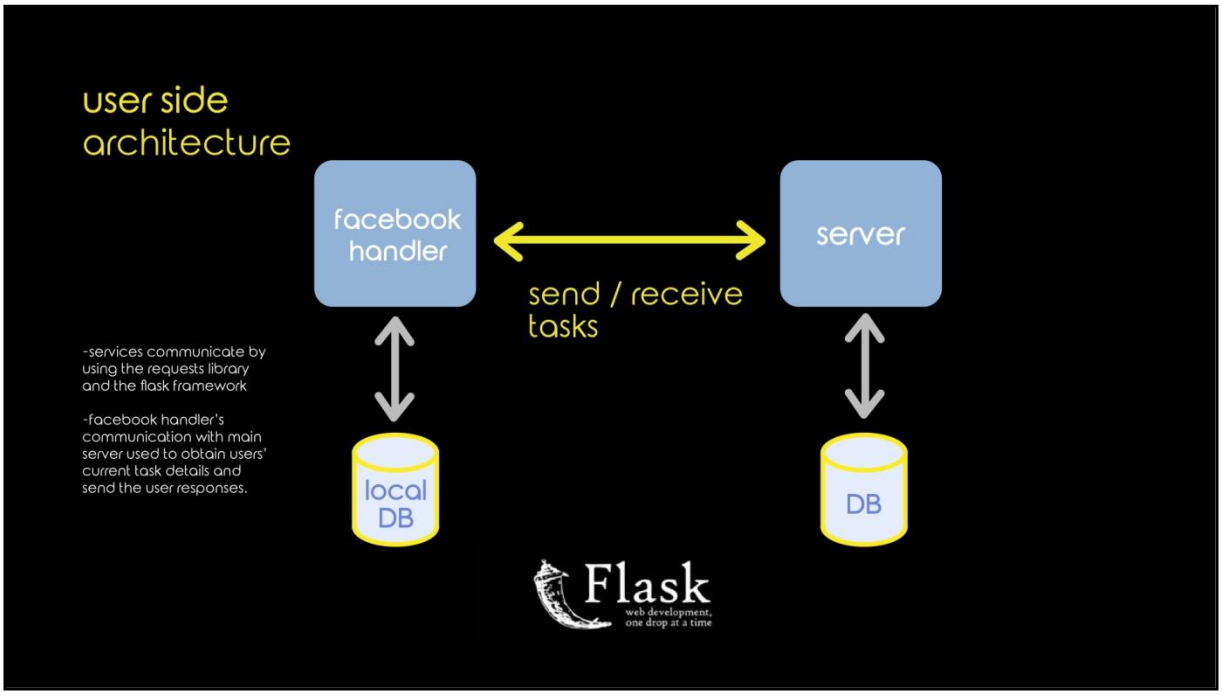


user side architecture

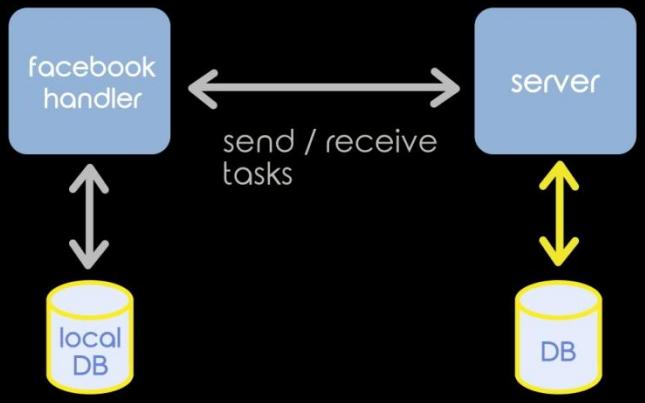


-the user can change his nickname and the coworker on the settings page


user side architecture
database:
_format: JSON
_hosting platform: mLab
services:
_language: python
_frameworks: Flask & messenger framework
-facebook handler responsible for creating html's trough templates with the information requested by the user.
-facebook's database contains the personas' scripts and the users' task progression.
facebook handler
server
send / receive tasks
local DB
DB
python
mLab
heroku
{JSON}


user side architecture
-services communicate by using the requests library and the flask framework
-facebook handler's communication with main server used to obtain users' current task details and send the user responses.
facebook handler
server
send / receive tasks
local DB
DB
Flask
web development, one drop at a time

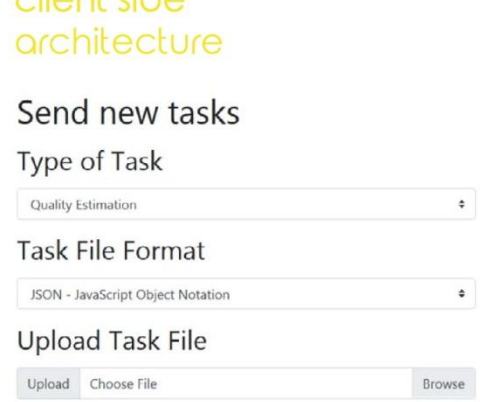
user side architecture



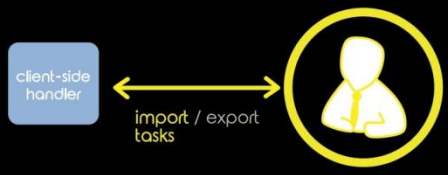
-main server responsible for analyzing the users' responses and the clients' new tasks.

-main server's database contains all the tasks and user information.

client side architecture



-the client can send tasks using a web page
-the tasks can be either quality estimation type or categorization type
-the tasks can be in format JSON, CSV (comma sepatated values) or TSV (tab separated values)



client side architecture

Tasks

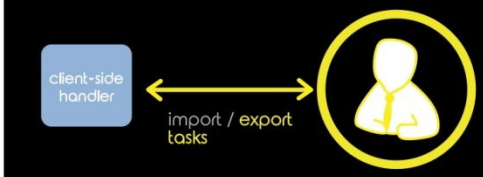
New Tasks

CSV - Comma Separated Values | Export All | Export Filtered

Time | Description | Response

Identifier	Type	Time			Completed	# Retries
		Submission	Response (average)	Completion (last response)		
0	category	2019-06-05T18:02:55.836672	0:00:11	N/A	False	0
1	qe_translate	2019-06-05T18:03:09.181573	0:00:09	N/A	False	0

-the client can view information on the tasks he sent using a web page
-this task information is split in three categories: time, description and response
-tasks can be filtered by these categories
-tasks can be exported to JSON, CSV or TSV files



client side architecture

Tasks

New Tasks

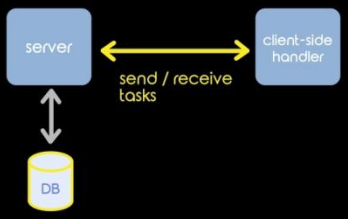
CSV - Comma Separated Values | Export All | Export Filtered

Time | Description | Response

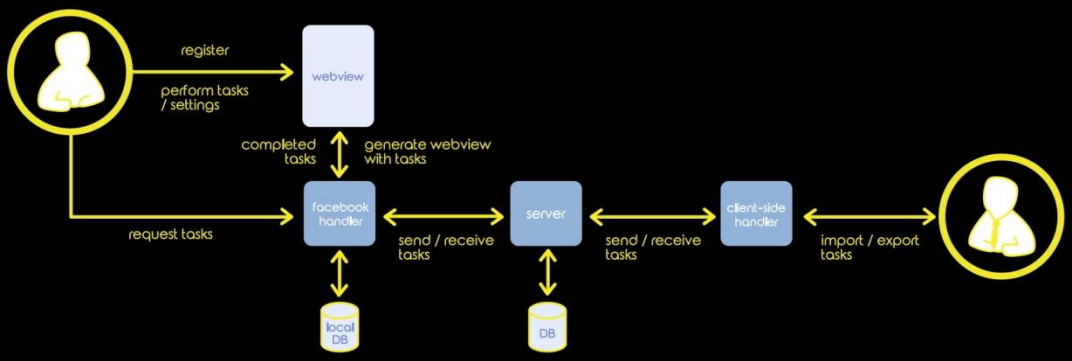
Identifier	Type	Time			Completed	# Retries
		Submission	Response (average)	Completion (last response)		
0	category	2019-06-05T18:02:55.836672	0:00:11	N/A	False	0
1	qe_translate	2019-06-05T18:03:09.181573	0:00:09	N/A	False	0

-client-side handler's communication with main server performed to obtain the client's tasks and its users' responses and to send new tasks imported by the client.

-task conversion and export done on the client's side to save the handler processing time.



architecture



future work

1. Payment
2. Notifications
3. Client side authentication


thank you!
TÉCNICO LISBOA
b a belas-artes ulisboa
Unbabel