How OSS Courses Prepare Developers for the Tech Industry

Sagi Amangeldi

Nazarbayev University, Astana, Kazakhstan
sagi.amangeldi@nu.edu.kz

Abstract. In the rapidly advancing world of technology, the significance of understanding the wider ecosystem of software development, especially the open source community, is paramount. This paper reflects on the experiences of participating in an Open Source Software (OSS) course at Nazarbayev University during the Spring semester of 2021. Focusing on engagement with the KISS Launcher, an open-source project for Android devices, the study sheds light on the insights gained regarding the software development lifecycle, community interactions, and problem-solving tactics within the OSS realm. This involvement acts as a microcosm of the broader software engineering industry, offering a glimpse into real-world challenges and equipping participants with not just technical skills, but an enriched understanding of collaborative dynamics and culture. Key learnings include proficiency enhancement in GitHub, improved communication acumen, technical prowess, especially in mobile development, and an appreciation for beta testing, among others. The paper underscores the pivotal role of OSS courses in bridging the gap between academia and the tech industry, preparing budding developers for the multifaceted world of software engineering.

Keywords: Open source software · Student experience · Computer engineering culture.

1 Introduction

In today's rapidly evolving technological landscape, software development is not merely about mastering a specific language or tool; it's about understanding the broader ecosystem in which software projects live and breathe. This environment, often dynamic and collaborative, plays a crucial role in shaping the practices, strategies, and methodologies employed in the tech industry [1]. At the nexus of this landscape is the open source community, a vast and intricate web of developers, projects, and collaborators dedicated to the shared use, modification, and distribution of software. Aspiring developers, aiming to immerse themselves in the tech industry, often find a solid foundational experience in Open Source Software (OSS) courses. Through my journey, I discovered that my involvement with OSS not only equipped me with technical skills but also provided a deep understanding of the culture, communication, and collaborative nuances that define the world of software engineering.

By delving into a real-world OSS project, I was able to gain firsthand insights into the challenges, problem-solving tactics, and community interactions that characterize the software development lifecycle. This experience, akin to an experiential learning journey, enabled a smooth transition into a professional software developer role. The similarities between the open source community's workings and the operations of software engineering companies became evident. From managing complex problems, and interacting with a global community on platforms like GitHub, to understanding the intricacies of project management and bug reporting, the OSS course provided a microcosm of the broader software engineering world.

This paper seeks to explore and elucidate the many ways in which engagement with open-source projects through formal OSS courses prepares developers for the challenges and opportunities of the tech industry based on my understanding achieved by attending the Open Source Software course taken in the Spring semester of 2021 at Nazarbayev University.

1.1 Course Description

The course's learning outcomes can be classified into two categories: theoretical knowledge and practical application. The former pertains to comprehending the principles of Open Source Software, including its history and current landscape, taught through lectures. Evaluation of this section was conducted via quizzes, a midterm, and an in-depth paper on a relevant open source topic. On the other hand, the practical aspect taught students how to actively engage in open source development. For this, students had to select a project, take on a specific role, make commitments to the project, and submit records of their contributions as reports. Regular brief updates on their commits and pull requests were expected, as were more comprehensive reports reflecting on their tasks and choices. This document is an extension of the course's concluding report, delving deeper into the contribution section.

1.2 Motivation and Project Selection

The endeavor of project selection was anchored by two principal criteria. Firstly, the essential was to select a project wherein the core concept was thoroughly comprehensible. Secondly, it was important to ensure that my contribution could substantially foster the advancement of the project. With a seasoned background in JavaScript and Java, reinforced by an internship that involved web and mobile application development, my preference naturally leaned toward projects predominantly using these languages. This proficiency not only delineated my expertise in front-end and mobile development but also emphasized the desirability of projects aligning with these languages.

While my exploration did lead me to other open-source projects like AmazeFileManage [2], Travel Mate [3], and Mozilla BugBug [4], their suitability was discounted owing to reasons ranging from a limited active contributor pool, as seen in AmazeFileManage, to the overwhelming influx of new contributors in

Mozilla BugBug, making meaningful contribution challenging. Additionally, the intricate complexities inherent in some projects were beyond my current proficiency, further solidifying the chosen project's alignment with my aspirations and capabilities. The project I selected is called the KISS Launcher [5].

2 Description of the Project

The KISS launcher [6], an open-source application developed for Android devices, exemplifies the embodiment of efficiency in design and functionality. Originating from the development platform, GitHub, it has found its presence on eminent app distribution platforms like Google Play Store and F-Droid.

The acronym 'KISS' normally is short for "Keep it Simple, Stupid"; the KISS project uses KISS to mean "Keep it Simple and Stupid". This appellation serves as a metaphorical representation of the launcher's minimalistic design and its intentional abstention from Internet connectivity. The launcher, adaptive in nature, progressively aligns with user patterns, enhancing search efficiency. This deviation from conventional, intricate launchers augments battery longevity and boosts device performance. A salient feature of this application is its swift search mechanism that spans across apps, contacts, and system settings. Furthermore, collaborative contributions have enriched the launcher, offering an expanded feature set that includes widget incorporation, instantaneous contact search, and enhanced configuration options.

2.1 The Leadership Paradigm

The project leadership of the KISS project exhibits a predominantly monarchical structure. This is exemplified by the pivotal role assumed by Matthieu Bacconnier (@Neamar) [7], the visionary and primary developer of the project. Since inception(at the time of contribution), the project has seen 169 releases, all orchestrated by Bacconnier. Quantitative insight into his contributions is evident from the 1,464 commits on GitHub, a number that starkly contrasts with the subsequent highest contributor's 127 commits (Fig.1). While Bacconnier remains the linchpin of the project, a consortium of contributors aids in diverse capacities such as bug resolution, testing, ideation, and linguistic translation. Notwithstanding, certain operational facets exhibit a democratic distribution amongst the project's stalwarts. Ancillary tasks like app uploads to distribution platforms and linguistic translation manifest a decentralized approach.

2.2 Governance Protocols

The KISS project embraces an inclusive ethos, welcoming contributors irrespective of their pedigree. While entry barriers are minimal, authority within the community is contingent upon one's contribution magnitude. Unlike rigidly structured projects like Linux [8], KISS mirrors the organic hierarchy observed in entities like the Apache group [9]. Role delineations within the community are reflective of the responsibilities they assume.

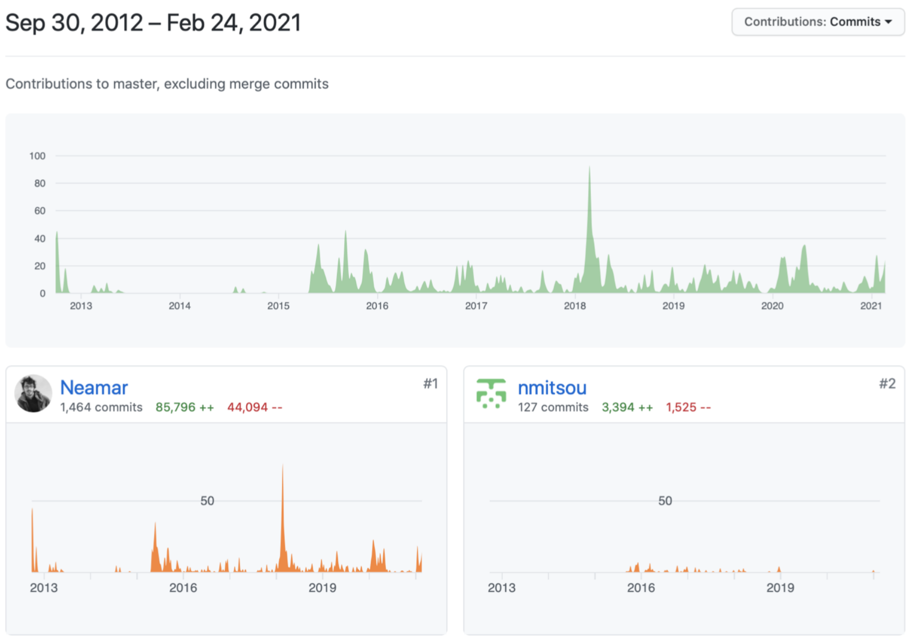


Fig. 1. Top contributors' commits.

2.3 Technical Dimensions

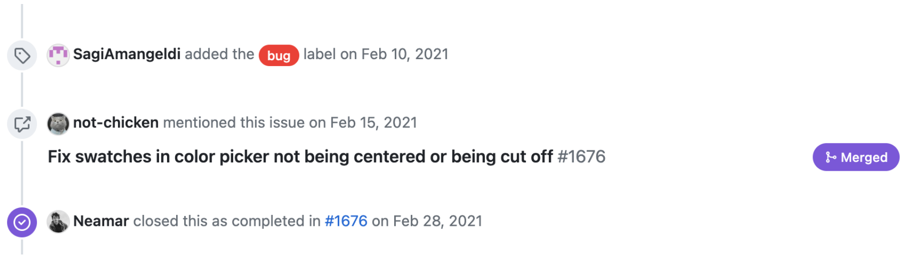
Predominantly scripted in Java (98.4%) with a marginal Python (1.3%) contribution, the KISS project aligns with the AGPLv3+ [10] copylefted libre software directive, underscoring open access and modification liberties. The architectural foundation was laid three years ago, with subsequent iterations primarily focusing on enhancements and bug resolutions. Its compactness and speed, indicative of an efficient structural design and algorithmic formulation, may present comprehension challenges for novices in Android development. The project supports 20 languages and for translation, this project uses the web tool 'weblate.com' [11].

3 Role and Work Done

3.1 Initial Contributions

Upon embarking on this project, my primary objective was the successful completion of the project rather than a specific role. Opportunities presented themselves for code development and requirement engineering in other tasks. However, fortuitously, my responsibilities aligned with beta testing. I was privileged to be a part of the beta test group, consisting of 20 members, which evaluated version 3.15.3 [12] of the software application. My consistent, daily assessments yielded an issue pertaining to the keyboard [13], which remains unresolved as of this writing. Also, evidence of my observations was corroborated by a video link.

Subsequent to my initial feedback, version 3.15.4 was launched. Post-update, the keyboard issue persisted, accompanied by two additional glitches: a malfunctioning ’back’ button [14] and a UI incongruence [15]. A pull request addressing this matter was later received (Fig. 2).


SagiAmangeldi added the bug label on Feb 10, 2021

not-chicken mentioned this issue on Feb 15, 2021

Fix swatches in color picker not being centered or being cut off #1676 Merged

Neamar closed this as completed in #1676 on Feb 28, 2021

Fig. 2. Initial accepted pull request.

Further scrutiny revealed another issue concerning an invisible application icon [16]. An additional discrepancy was identified in the incorporation of icon packs, leading to application crashes. This issue has been duly reported but remains pending review [17].

3.2 Other Contributions

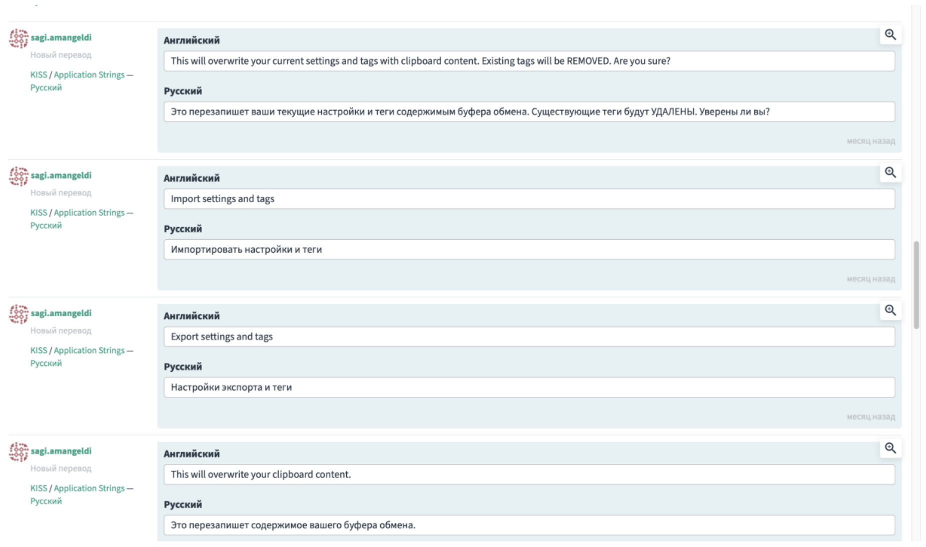
In the description of the project, there was an expressed need for linguistic assistance within the project. Beyond my primary responsibilities, I facilitated the project by rectifying translation issues and addressing complex string anomalies. Initial evaluations indicated that the Russian translation was incomplete, with merely 68% accuracy. After a two-week collaboration, we achieved comprehensive translation [18]. Figure 3 serves as verification.

Drawing from my expertise in Russian translation, I spearheaded the translation of the project into Kazakh to cater to a wider audience [19].

3.3 Conclusion of Contributions

In concluding this discourse, I wish to elucidate the profound insights and knowledge I have garnered throughout the duration of this course. Engaging with Open Source Software (OSS) projects proved to be a stimulating experience. My enthusiasm for the work was consistent, as evidenced by my proactive involvement from the outset.

Over the subsequent 2.5 months, I have contributed to the KISS project encompasses 31 commits within this timeframe [20]. These contributions span


sagi.amangeldi
Новый перевод
KISS / Application Strings — Русский

Английский
This will overwrite your current settings and tags with clipboard content. Existing tags will be REMOVED. Are you sure?

Русский
Это перезапишет ваши текущие настройки и теги содержимым буфера обмена. Существующие теги будут УДАЛЕНЫ. Уверены ли вы?

месяц назад

sagi.amangeldi
Новый перевод
KISS / Application Strings — Русский

Английский
Import settings and tags

Русский
Импортировать настройки и теги

месяц назад

sagi.amangeldi
Новый перевод
KISS / Application Strings — Русский

Английский
Export settings and tags

Русский
Настройки экспорта и теги

месяц назад

sagi.amangeldi
Новый перевод
KISS / Application Strings — Русский

Английский
This will overwrite your clipboard content.

Русский
Это перезапишет содержимое вашего буфера обмена.

Fig. 3. weblate.org detailing translation history.

across addressing 5 bug reports, rectifying 1 crash, incorporating 3 enhancements, proffering 4 supportive remarks to fellow contributors, making 3 endeavors to resolve longstanding bug issues, elevating the Russian translation from 68%, and effectuating a comprehensive translation along with the implementation of string functions for the Kazakh language.

In a reflective analysis of the KISS project's contributors ranking, as portrayed in Figure 5, I am positioned at the 15th rank [20]. While I acknowledge that such rankings may not encapsulate the full extent of one's contributions, securing the 15th rank amongst 195 contributors within a span of 2.5 months undeniably stands as a noteworthy accomplishment in my scholarly journey.

4 Lessons learned

Participation in Open Source Software (OSS) projects can be construed as a robust pedagogical tool that enables students to hone their technical competencies. Through my immersion in this domain, I discerned that engagement with OSS serves as a pivotal catalyst. The course elucidated the intricate processes underlying the conception and execution of large-scale projects, strategies for effective communication within OSS initiatives, the nuances of making meaningful contributions, and adeptly navigating technical impediments. My satisfaction with the project selection was a testament to the dividends of the dedication expended throughout the duration of the course.

Furthermore, it warrants emphasis that the initial project selection can profoundly influence the trajectory of a contributor's future endeavors, given that

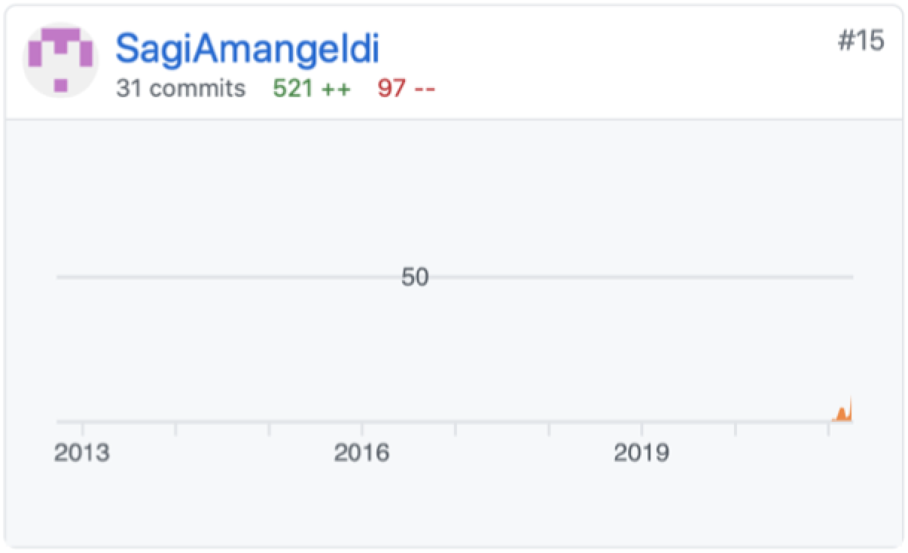


Fig. 4. Commits on the KISS project.

the inaugural forays into the OSS sphere are central. Identifying a project complemented by a conducive environment and collaborative contributors is illustrative for those aspiring to make impactful contributions. I was fortuitously aligned with a project characterized by an amicable lead and collaborative co-contributors. For neophytes in the domain, it's important to strike a balance between the zeal for unraveling novel technological facets pertinent to the project and ensuring their contributions remain germane.

Adopting a perspective that mitigates potential challenges in OSS engagement can be invaluable. Moreover, fostering a culture of inquiry and acknowledging the inevitability of errors underpins the learning ethos of this course.

In reflection, I wish to underscore the principal domains wherein I witnessed considerable enhancement:

GitHub Proficiency: While I had prior familiarity with GitHub, my engagement deepened my understanding, elevating me to a more proficient user.

Communication Acumen: Collaborating with the project lead and an expansive network of contributors enriched my insights into the dynamics of large-scale project management and effective communication strategies within OSS frameworks.

Technical Prowess: My technical repertoire expanded significantly. My endeavor to contribute as a code developer necessitated a deep dive into mobile development and Java, with a particular focus on Android and the architecture of the KISS project. Additionally, my engagement with weblate.org bolstered my proficiency with translation and string formulations. I grasped the intricacies of string interconnections and the overarching translation protocols for expansive projects like ours.

Additional Insights: I garnered an appreciation for Beta testing mechanisms, and gained insights into licensing protocols, and their paramount significance. I acquired foundational knowledge about sponsorship mechanisms, particularly through platforms like liberapay.com. My exposure also encompassed the nuances of contributing to updates, and understanding release processes, among other salient learnings not exhaustively enumerated herein.

5 OSS Courses: Bridging Academia and Tech Industry

The dynamic and collaborative realm of software development transcends the mere grasp of programming languages and tools. Central to this ecosystem is the open source community, a confluence of developers, projects, and collaborators, all geared towards the shared ideology of open, modifiable, and distributable software. The relevance of Open Source Software (OSS) courses, such as the one undertaken at Nazarbayev University in 2021, cannot be understated in this context. These courses not only impart technical know-how but also acclimate students to the multifaceted culture, communication methodologies, and collaborative dynamics that permeate the software engineering arena. A deep dive into OSS projects, as seen in the KISS Launcher experience, elucidates real-world challenges and problem-solving strategies, while simultaneously highlighting the essence of community interactions and software lifecycle intricacies. Such engagement, emblematic of experiential learning, serves as a bridge, facilitating a seamless transition from academic settings to professional developer roles. The parallels between open source community operations and those of professional software companies become increasingly palpable. Be it grappling with intricate problems, interfacing with a diverse global community on platforms such as GitHub, or mastering the subtleties of project management and bug reporting, the realm of OSS provides an invaluable preparatory ground for emerging developers. With this foundation, aspirants are better positioned to navigate the ever-evolving challenges and opportunities of the tech industry.

5.1 Personal Experience from Today's Perspective

This contribution was made 2 years ago as a senior student. After graduation from university, I worked remotely for 1 year as a web developer and now I am working as a middle software engineer. Having actively participated in an open-source project I was armed with a unique set of experiences that greatly informed and enriched my contributions in a structured corporate setting. One of the first things that struck me was how integral the process of documentation was in the open-source realm. With developers from varying backgrounds and time zones contributing to a project, clear and concise documentation becomes the unifying language. For instance, as was said above while working on a particular module in the OSS project, I took the initiative to make a detailed documentation of existing bugs. This wasn't just about explaining what the code did, but more about why certain decisions were made, potential pitfalls, and how future contributors might extend or modify it. This forward-thinking approach was something I carried with me into the software engineering role. When our company decided to refactor a legacy system, my well-documented code served as a roadmap, making the transition process smoother and reducing the onboarding time for new team members.

Furthermore, the open-source ecosystem taught me that even the most minute change in code is accompanied by a rigorous process. I vividly recall a situation where a simple update to a function, which I initially perceived as trivial, required multiple layers of review and validation. Each line of code was scrutinized, and tested in multiple environments, and even the potential downstream implications were assessed. Even though I did not contribute as a coder, this meticulous approach was something I deeply internalized. Later, when working on a critical feature at the software company, I instigated a thorough review and testing protocol, ensuring that our deployment was robust and free from unforeseen repercussions.

Moreover, the iterative process of open-source contribution, where peer reviews are fundamental, sharpened my ability to write clear, maintainable code. I recall a time in the open-source project when a simple feature that I suggested had multiple revisions, with inputs from different contributors, leading to a more efficient and cleaner version of the function. Later in the software company, during a critical product launch, this skill proved instrumental. I was able to quickly draft code that not only met the functional requirements but was also readily understandable by my peers, reducing the lead time to product deployment.

6 Conclusion

In the rapidly evolving software development landscape, the open-source community stands as a crucial touchstone for budding developers. OSS courses, like the one at Nazarbayev University, play an instrumental role in bridging the theoretical world of academia and the hands-on environment of the tech industry. These courses immerse students in the ethos of collaborative development, underscoring the significance of community interactions and software life cycle intricacies. Personal experiences, such as those shared, underscore the importance of thorough documentation, rigorous code review, and the iterative process foundational to open-source contributions. These attributes, cultivated through active participation in OSS projects, translate seamlessly into the corporate world. In this setting, well-documented codes become the backbone of efficient system transitions, while meticulous testing and review protocols ensure the robustness of deployments. Furthermore, the skills honed through continuous peer reviews in the OSS environment lead to the creation of clear, maintainable code, expediting the product deployment cycle in professional settings. In essence, the open-source realm offers a comprehensive preparatory platform, equipping emerging developers with the necessary tools and perspectives to thrive in the ever-adaptive tech industry.

Acknowledgements. Professor Antonio Cerone was a lecturer at the university course introducing Open Source.

References

1. Lundell, B., Persson, A., Lings, B.: Learning through practical involvement in the OSS ecosystem: experiences from a masters assignment. In: Feller, J., Fitzgerald, B., Scacchi, W., Sillitti, A. (eds.) OSS 2007. ITIFIP, vol. 234, pp. 289–294. Springer, Boston, MA (2007). https://doi.org/10.1007/978-0-387-72486-7_30
2. AmazeFileManager Repository. https://github.com/TeamAmaze/AmazeFileManager. Accessed 25 Jan 2021
3. Travel Mate Repository. https://github.com/project-travel-mate/Travel-Mate. Accessed 25 Jan 2021
4. Mozilla BugBug Repository. https://github.com/mozilla/bugbug. Accessed 25 Jan 2021
5. KISS Launcher Official Site. https://kisslauncher.com/. Accessed 25 Jan 2021
6. KISS Launcher Repository. https://github.com/Neamar/KISS. Accessed 25 Jan 2021
7. Matthieu Bacconnier GitHub page. https://github.com/Neamar. Accessed 25 Jan 2021
8. Linux Official Site. https://www.linux.org/. Accessed 25 Jan 2021
9. Apache Official Site. https://www.apache.org/. Accessed 25 Jan 2021
10. KISS License File. https://github.com/Neamar/KISS/blob/master/LICENSE. Accessed 25 Jan 2021
11. Weblate - Web-based continuous localization. https://weblate.com. Accessed 25 Jan 2021
12. KISS Launcher, Pre-release of 3.15.3 version. https://github.com/Neamar/KISS/releases/tag/v3.15.3
13. Sagi Amangeldi: Keyboard issue. https://github.com/Neamar/KISS/issues/1662
14. Sagi Amangeldi: 'Back' button issue. https://github.com/Neamar/KISS/issues/1664
15. Sagi Amangeldi: Colors are not centered. https://github.com/Neamar/KISS/issues/1671
16. Sagi Amangeldi: Invisible icon of one application in the favorites. https://github.com/Neamar/KISS/issues/1684
17. Sagi Amangeldi: Crash happens when you change the icon pack. https://github.com/Neamar/KISS/issues/1719
18. KISS Launcher - weblate.org page. https://hosted.weblate.org/changes/browse/kiss/strings/ru/?page=8
19. KISS Launcher - weblate.org page. https://hosted.weblate.org/changes/browse/kiss/strings/kk/?page=3
20. Sagi Amangeldi GitHub page.https://github.com/SagiAmangeldi
21. KISS Launcher, Contributors. https://github.com/Neamar/KISS/graphs/contributors