Design and Implementation of a Modern Standard Marine Communication Phrases (SMCP) Language Learning Application

Degree project report in Computer Science

Mirco Ghadri & Zakariya Omar

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2023
www.chalmers.se

Degree project report 2023

Design and Implementation of a Modern Standard Marine Communication Phrases (SMCP) Language Learning Application

Mirco Ghadri & Zakariya Omar

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2023

Design and Implementation of a Modern Standard Marine Communication Phrases (SMCP) Language Learning Application
Mirco Ghadri & Zakariya Omar

Project Supervisor: Annamaria Gabrielli, Fackspråk och Kommunikation
Supervisor: Morteza Abdipour, Computer Science and Engineering
Examiner: Lars Svensson, Computer Science and Engineering

Degree project report 2023
Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone +46 31 772 1000

Cover: Marine Deck Officer or Chief mate on deck of vessel or ship . He holds VHF walkie-talkie radio in hands. Ship communication

Typeset in LaTeX
Gothenburg, Sweden 2023

Design and Implementation of a Modern Standard Marine Communication Phrases (SMCP) Language Learning Application
Mirco Ghadri & Zakariya Omar
Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg

Abstract

This Bachelor's thesis describes the development of a modern and updated Standard Marine Communication Phrases (SMCP) language learning application for the shipping and marine technology division at Chalmers University of Technology. The aim of this project is to design and construct a functional demo application that serves as a proof of concept and provides valuable insights for future development.

By conducting data gathering, iterative design prototyping, and software implementation, the project successfully accomplished its goals. The outcomes include the identification of user needs and the creation and evaluation of a design prototype. Online surveys were utilized for data gathering and evaluation, allowing for the collection of user feedback and insights.

The project resulted in the development of a fully functional cross-platform application compatible with Web, iOS, and Android platforms. The evaluation process validated the effectiveness of the design, while also highlighting potential areas for future enhancement. Ultimately the project's findings and insights have the potential to significantly impact SMCP learning at Chalmers University of Technology and beyond.

Keywords: Standard Marine Communication Phrases (SMCP), Language Learning Application, Ship Communication, Cross-platform Application, User-centered Design, Software Development

Acknowledgements

We would like to express our sincere appreciation to our supervisor, Morteza Abdipour, for his invaluable guidance and support throughout this project. His expertise and advice have been instrumental in shaping our work. We would also like to extend our gratitude to our project supervisor, Annamaria Gabrielli, for proposing the thesis and guiding us along the way. Her continuous support and valuable insights have contributed to the success of this project. Her strong commitment to this thesis and effort in distributing our surveys to students and teachers internationally have made this project possible.

Mirco Ghadri & Zakariya Omar, Gothenburg, June 2023

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

SMCP	Standard Marine Communication Phrases
IMO	International Maritime Organization
UCD	User-centered Design
UI	User Interface
VSC	Visual Studio Code
IDE	Integrated Development Environment

Contents

List of Acronyms **ix**

List of Figures **xiii**

1 Introduction **1**
1.1 Background 1
1.2 Purpose 1
1.3 Goal 2
1.4 Scope 2

2 Methodology & Project Approach **5**
2.1 Agile Project Management 5
2.1.1 Scrum 5
2.2 User-centered Design 5

3 Technical Background **7**
3.1 User Interface(UI) 7
3.1.1 Wireframe 7
3.1.2 Figma 7
3.2 Cross-platform Development 7
3.2.1 Flutter (Framework) 8
3.2.1.1 Dart (Programming Language) 8
3.2.1.2 Flutterflow (Low-code Builder) 8
3.2.2 Visual Studio Code (Code editor) 8
3.3 Version Control 8
3.3.1 Git (Version Control System) 8
3.3.2 GitHub (Code hosting platform) 9

4 Method **11**
4.1 Design 11
4.1.1 Data gathering 11
4.1.1.1 Survey 11
4.1.2 Prototyping 12
4.1.2.1 Wireframe prototyping 12
4.1.2.2 Figma prototyping 12
4.1.3 Evaluation 12
4.1.3.1 Survey 13

4.2 Software Implementation 13
4.2.1 Flutterflow 13
4.2.2 Manual coding 14

5 Results 15
5.1 Data gathering 15
5.1.1 Survey results(students) 15
5.1.2 Survey results(teachers) 18
5.1.3 Survey Analysis & Findings 19
5.2 Prototyping 21
5.2.1 Wireframe 21
5.2.2 Figma prototype 23
5.3 Evaluation 27
5.3.1 Survey results(students) 27
5.3.2 Survey Analysis and Findings 37
5.4 Software implementation 38
5.4.1 Flutterflow 38
5.4.2 Home menu 40
5.4.3 Glossary 43
5.4.4 Games 46
5.4.4.1 Quiz 47
5.4.4.2 The Ship Game 50
5.4.4.3 Missing Word Game 53
5.4.5 Dictionary 54

6 Discussion 57
6.1 Flutterflow 57
6.1.1 Limitations 57
6.1.2 Bugs 58
6.1.3 Flutterflow Support & Community 59
6.2 Ethical & Ecological Aspects 60
6.3 Future Work 61

7 Conclusion 63
7.1 Design 63
7.2 Software Implementation 63
7.3 Summary 64

Bibliography 65

A Appendix: Custom code I
A.1 The Ship Game I
A.2 Voice Recording functionality III
A.3 Recording playback functionality IV
A.4 Recording Microphone IV
A.5 Search filter VI
A.6 Quiz answer options shuffling VI

List of Figures

1 Resource availability 15
2 Skill relevance 16
3 Feature usefulness 16
4 Additional Features(students) 17
5 Skill difficulty 18
6 Suitable platform 18
7 Additional features(teachers) 19
8 Wireframe home page 21
9 Wireframe quick menu 21
10 Wireframe practice page 22
11 Wireframe missing word game 22
12 Figma home page 23
13 Figma quick menu page 23
14 Figma glossary menu page 24
15 Figma glossary practice page 25
16 Figma game menu page 26
17 Figma missing word game page 27
18 Number of icons 28
19 Colors and shading 29
20 Button and icons intuitiveness 30
21 Full page or half page 31
22 Search bar location 32
23 The Ship Page user friendliness 33
24 Page navigation and ease of use 34
25 Phrase completion steps difficulty 35
26 Perceiving completion of phrase difficulty 35
27 Prototype readiness 36
28 Homepage in Flutterflow workspace 38
29 Splash Screen presented to user when starting the application 39
30 Home page created in Flutter 39
31 Home menu in the Flutter application 40
32 Settings - light mode 41
33 Settings - dark mode 41
34 Homepage Flutter in Dark Mode 42
35 Home menu in Flutter Dark mode 42
36 Glossary menu page 43

37 General terms practice page in Glossary 44
38 Microphone pressed 45
39 Enabled playback button 45
40 Letters practice page in Glossary 45
41 Digits practice page in Glossary 45
42 Games menu page 46
43 Quiz first question 47
44 Quiz correct answer selected 48
45 Quiz wrong answer selected 48
46 Quiz continue button & text hidden 49
47 Quiz question & options hidden 49
48 Quiz completion page 50
49 The Ship Game page 1 51
50 The Ship Game page 2 - Game page 52
51 The Ship Game - Game page checked 52
52 Missing word game 53
53 Missing word game checked 53
54 Dictionary 54
55 Dictionary filtered with search 54
56 Homepage search autocomplete 55
57 Dictionary redirect 55

1
Introduction

1.1 Background

The maritime industry is a constantly expanding global business that depends on substantial international collaboration in order to keep developing, so as to meet expanding global market needs. Safety, in particular on vessels, is crucial for this industry to thrive. Safety on vessels, for crew and cargo, is primarily dependent on the good communication skills of both shore and sailing personnel. Consequently, the International Maritime Organisation (IMO), a UN organ established in 1948, with currently 173 member states [1], has issued a number of regulations and standards with regard to the seafarers' communication skills, but also with reference to the 'Lingua Franca of the sea'. Today, Maritime English is the only language for specific purposes in the world, that is reinforced by international legislation.

At Chalmers University of Technology, all Bachelor's studies within shipping and marine technology include learning activities and learning outcomes with reference to communication skills, particularly Standard Marine Communication Phrases (SMCP). According to the IMO convention, all officers in management and operational positions onboard must be 'conversant with SMCP' at different levels [2], or else they will not be able to sail international waters. This means that all officers in operational and management positions must be able to carry out a complete conversation using only SMCP.

At Chalmers, the significance of effective communication skills, specifically SMCP, is recognized within the shipping and marine technology programs. Since 2007, a desktop application for SMCP learning has been available exclusively on Chalmers computers with good results. However, to better accommodate the evolving educational landscape and provide students with greater accessibility, there is a growing need for a more modern and updated application that can be readily accessed beyond the confines of school and the restrictions of desktop computers.

1.2 Purpose

The purpose of this project is to address the growing need for a modern and updated application for Standard Marine Communication Phrases (SMCP) learning at Chalmers University of Technology.

1.3 Goal

The goal of this project is to lay the foundation for the development of a more modern and updated Standard Marine Communication Phrases (SMCP) learning application at Chalmers University of Technology. The specific objectives of this project are as follows:

1. **Identification:** Identify the requirements and key features necessary for a more modern and effective SMCP learning application. This involves conducting research, gathering input from the target user group and analyzing the limitations of the current desktop application.

2. **Design:** Design a comprehensive application that aligns with the identified requirements and addresses the limitations of the existing desktop application.

3. **Evaluation:** Evaluate the design of the application by gathering feedback from the targeted user group. This evaluation process will provide insights into the application's effectiveness, user satisfaction, and areas for improvement. Based on the findings, revisions and refinements will be made to optimize the application's design.

4. **Implementation:** Implement the finalized design into a working application that incorporates the identified requirements and design improvements. This phase will involve developing the necessary software and integrating the learning materials.

By addressing these 4 objectives, this report intends to lay the foundation for the development of an application that fulfils the educational needs of students and equips them with the necessary communication skills for success in the maritime industry.

1.4 Scope

The scope of this project is to develop a working demo application with a focus on the structure and functionality of the application, rather than delivering a comprehensive end product. The demo application will provide the following features:

- **Listening:** Users can listen to a native speaker's pronunciation of the SMCP phrase

- **Recording:** Users can record themselves reading the SMCP phrase.

- **Pronunciation Comparison:** Users can compare their own pronunciation with the native speaker's.

The demo application will include a selected sample of SMCP phrases to effectively demonstrate its functionality. It will not encompass all SMCP phrases. The priority of this project is to establish the foundation and demonstrate key features, leaving room for future additions and expansions.

By focusing on the structure and functionality of the application, this project sets the groundwork for further development and refinement in the future.

2
Methodology & Project Approach

In this section, we will provide an overview of the taken project approach and the methodologies employed to achieve the project's objectives.

2.1 Agile Project Management

Agile project management was adopted as the primary methodology for this project. Agile is a flexible and iterative approach that emphasizes collaboration, adaptability, and continuous improvement [3]. By embracing Agile principles, we aimed to enhance communication, increase productivity, and deliver high-quality results. Within the Agile framework, the Scrum methodology was specifically utilized.

2.1.1 Scrum

Scrum is an agile framework that divides a project up into smaller sub-projects called sprints. The work is reviewed at the end of each sprint, revisions are made for the next sprint and the process is repeated until the project is complete [4].

By employing Scrum, we organized the work into weekly sprints. During each sprint, we held regular meetings to discuss progress, tackle challenges and ensure alignment. At the end of each sprint, a sprint review and retrospective were conducted to assess the deliverables, gather feedback, and identify opportunities for improvement. The adoption of Scrum facilitated a transparent and collaborative work environment. This iterative approach ensured a more efficient and responsive development process.

2.2 User-centered Design

User-centred design (UCD) was another methodology employed in this project. UCD focuses on understanding the needs, preferences, and behaviours of the end-users throughout the design and development process [5]. By incorporating UCD principles, we conducted user research, gathered user feedback and iteratively refined the application's design based on user insights gained from evaluation. This approach helped ensure that the application was intuitive, user-friendly, and effectively met the needs of its intended users.

By combining Agile project management with user-centred design, we aimed to create an iterative and user-centric development process. These methodologies provided a structured framework for efficient collaboration, frequent feedback, and continuous improvement, ultimately leading to the successful achievement of the project's objectives.

3
Technical Background

In this section, we will delve into the underlying technologies, frameworks, and concepts that are relevant to this project.

3.1 User Interface(UI)

The User Interface (UI) is the bridging gap between the user and the underlying functionality of an application. UI refers to how a user interacts with an application from the visual and interactive elements, such as buttons and menus, that enable interaction. It includes the design, layout, and overall functionality of everything that the user interacts with on the interface [6]. In the development of a detailed UI, two important tools come into play.

3.1.1 Wireframe

Wireframes are usually developed in the initial phases of the design process and help designers outline their ideas into a visual representation of the application's UI, usually in a two-dimensional skeleton showcasing what they'll include in the application. It serves as a blueprint for the overall structure and functionality of the application and can be either drawn on paper or digitally [7].

3.1.2 Figma

Figma is a web-based design platform that offers robust design, prototyping, and code-generation tools. It enables users to create detailed designs and interactive prototypes. Designing complex, detailed and interactive UIs is one of its many applications, and it even allows for collaboration in real-time [8].

3.2 Cross-platform Development

Cross-platform development is the process of building an application from a single codebase that can run on multiple operating systems or platforms. Rather than develop a specific application for each system or platform, developers can code once and deploy it everywhere. This saves both time and resources [9]. There are several tools and frameworks available for cross-platform development.

3.2.1 Flutter (Framework)

Flutter is an open-source software development framework developed by Google . It enables the development of cross-platform applications for mobile, web and desktop from a single codebase. Flutter allows for the creation of both high-performance and visually appealing applications [10].

3.2.1.1 Dart (Programming Language)

Dart is an object-oriented and open-source programming language developed by Google. It is the programming language utilized by Flutter and has a syntax which is similar to Java [11]. Dart can compile to ARM and x64 machine code, but can also be transpiled to Javascript, explaining why Flutter apps can run on all platforms, even web [12].

3.2.1.2 Flutterflow (Low-code Builder)

Flutterflow is a third-party web-based visual builder tool for the Flutter framework. Being a low-code visual builder, it enables the creation of a detailed user interface for Flutter applications with little to no coding as a developer and with faster development time. Flutterflow supports the addition of manual Dart code in the forms of custom functions, actions and widgets [13].

3.2.2 Visual Studio Code (Code editor)

Visual Studio Code (VSC) is a powerful and versatile desktop and web-based code editor with many functions of that of an Integrated Development Environment (IDE). It comes with a range of different features, including IntelliSense code auto-completion, debugging and version control integration [14]. VS Code supports many different programming languages, frameworks and extensions, the Flutter framework being one of them. Integrating perfectly with Flutter, it allows the user to effectively build, run & debug their Flutter application [15].

3.3 Version Control

Version control is a system for tracking and managing software code. It allows changes in the code to be tracked and different versions to be managed, and not only helps to maintain the integrity of the code but also provides a way to organize and collaborate [16]. Git and GitHub are two prominent tools in the field of of version control.

3.3.1 Git (Version Control System)

Git is an open-source distributed version control system originally developed by Linus Torvalds in 2005. Since then, it has become the most widely used Version Control System. Its distributed nature allows users to use it locally without access to the internet. Git also integrates flawlessly with most IDEs, including Visual Studio Code, to allow for a seamless coding experience [17].

3.3.2 GitHub (Code hosting platform)

GitHub is a cloud-based hosting platform that incorporates Git as its underlying version control system. It provides a hosting service in the form of a centralized repository where developers can store, manage, and collaborate on their code. With GitHub, users can easily share their projects, track changes, and leverage collaborative features such as pull requests and issue tracking [18].

4 Method

This chapter covers the steps taken to achieve this projects goal, from the data collection and designing phase, up until the software implementation phase.

4.1 Design

This project emerged from the necessity of updating an outdated application. To address this need and develop an application that not only fulfils the current requirements but also meets the current needs of users, a user-centred design approach was adopted. This approach emphasizes the importance of understanding the users, their goals, and their preferences in order to create a tailored and effective application.

In this section, we will present the key phases of the design process, including data gathering, prototyping, and evaluation.

4.1.1 Data gathering

In an attempt to accurately identify the needs of the application's targeted user group and create a positive user experience, user-centred data-gathering methods were employed in this project prior to initiating both the design and software development phases.

The aim was to gather the relevant information straight from the users in order to inform and guide the decision-making process and help ensure the application aligns with the user needs. The primary data collection tools employed were surveys.

4.1.1.1 Survey

Two surveys were developed; one aimed at teachers and one at students. Both surveys were distributed internationally to participants in the relevant field. The surveys were created using google forms and took a form of a mix of multiple-choice questions, open-ended questions and linear scales (1-5) questions. Providing both qualitative and quantitative answers that gave a more holistic understanding of the user's perspective. The teacher survey consisted of a total of 20 questions, including both main questions and follow-up questions. The student survey comprised 30 questions, including both main questions and follow-up questions. However, due to the branching nature of the survey, the average user was expected to answer approximately 20 questions. The survey was designed to address the following key questions and gather insights in the following areas:

- User Needs and Preferences
- User Interface Design
- Feature Prioritization
- Platform Accessibility
- User Feedback and Suggestions

The surveys were distributed internationally to ensure a diverse representation of participants from different educational institutions and cultural backgrounds. This approach enabled us to capture a wide range of perspectives and cater to the application's potential user base beyond Chalmers University of Technology. The choice of targeting students and teachers was deliberate, considering their involvement in maritime education and their direct engagement with SMCP learning. By including both groups, we aimed to gather a comprehensive understanding of the user needs, preferences, and challenges associated with SMCP learning.

4.1.2 Prototyping

Having analyzed the user data, a user-centred approach was taken in visualizing the findings and designing the resulting prototype. Starting with a wireframe design and advancing to the development of a Figma prototype.

4.1.2.1 Wireframe prototyping

Wireframes helped design the basic structure, layout and functionality of the application. Serving as a low-fidelity prototype that helped visualized the results from the findings and allowed for its refinement before the next stage of the prototyping phase. The wireframes were drawn by hand on paper.

4.1.2.2 Figma prototyping

Building on the wireframe, a more interactive and high-fidelity prototype was developed using the digital designing tool Figma. The Figma prototype expanded on the wireframe design and incorporated and allowed for greater detail and interactiveness than could be achieved with the wireframe.

4.1.3 Evaluation

In order to appraise the effectiveness and usability of the high-fidelity prototype, an evaluation was conducted. The evaluation served as a tool for validating the design decisions and functionalities implemented in the prototype. The primary data collection tool employed was a survey.

We evaluated the prototype to identify the strengths and weaknesses of the design, validate its functionality, and gather suggestions for enhancements. The feedback received from participants played a crucial role in informing iterative design improvements and shaping the final version of the prototype.

4.1.3.1 Survey

A survey was developed following a similar format to the previous surveys mentioned in this report. It was distributed internationally to participants in the relevant field and specifically targeted students to ensure that their perspectives and experiences were effectively captured. The survey was designed to address and gather insights into the following areas:

- **Design:** Evaluating the visual appeal and aesthetics.
- **Layout:** Evaluating the organization and arrangement of different elements.
- **Workflow:** Evaluating the usability and functionality.
- **User Feedback and Suggestions:** Gathering feedback and suggestions from participants on their overall experience with the prototype.

The survey was structured according to these areas with each section containing a set of questions that directly addressed the corresponding aspect of the prototype. By structuring the survey in this way, we aimed to systematically evaluate the design, layout, and workflow of the application, while also gathering user feedback and suggestions.

4.2 Software Implementation

The software implementation phase involved the actual development of the application, transforming the design concepts and user requirements from a Figma prototype into a functional and interactive application. This section provides an overview of the two primary approaches used in the implementation process: Flutterflow and manual coding.

4.2.1 Flutterflow

Flutterflow played a significant role in the implementation of the application's user interface. With its user-friendly drag-and-drop builder and extensive library of components, Flutterflow enabled a quicker prototype development than strictly manual coding on its own would have permitted. Its user-friendly drag-and-drop builder allowed for a more natural and seamless translation of our Figma design into an application user interface. By taking care of most of the design aspects using a visual drag-and-drop builder, it allowed us to focus more on the functionality of the application.

4.2.2 Manual coding

While Flutterflow's visual builder proved instrumental in the user interface (UI) development, manual coding was necessary to implement some of the application's core features, logic and functionality. The manual coding utilized the Dart programming language and the Flutter framework.

Manual coding addressed specific requirements that could not be accomplished through low-code methods alone. Some of the key areas where manual coding was essential were:

- **Customization of the application's behaviour:** Manual coding allowed us to tailor the application's behaviour to meet the specific needs and preferences of the users. This included implementing custom interactions, animations, games and user workflows that went beyond the capabilities of low-code tools.

- **Integration of external libraries:** Manual coding facilitated the integration of external libraries into the application. This allowed us to leverage additional functionality and services, incorporating advanced features that were not available out of the box. This included microphone recording functionality, which was not built into Flutter and therefore did not exist in Flutterflow (see 38 & A.2)

- **Implementation of complex features:** Manual coding was crucial for implementing complex features that required custom logic and intricate implementation. One such example was the Ship Game(see 5.4.4.2 & A.1).

Flutterflow provided the option for adding manual code in multiple forms. The 3 main forms were custom functions, custom widgets and custom actions. Although Flutterflow provided the ability for adding manual code, some of the manual code was added outside of Flutterflow, by coding in Visual Studio Code and uploading/pushing the code to the GitHub repository. The reason was that Flutterflow's manual coding editor had some limitations when it came to what was possible to code (see 6.1.1).

By combining the strengths of Flutterflow for UI development and manual coding for implementing intricate functionality, the implementation phase achieved a balance between efficiency and customization. This approach allowed for the efficient creation of a visually appealing and user-friendly application while accommodating the unique needs and complexities of the project.

5 Results

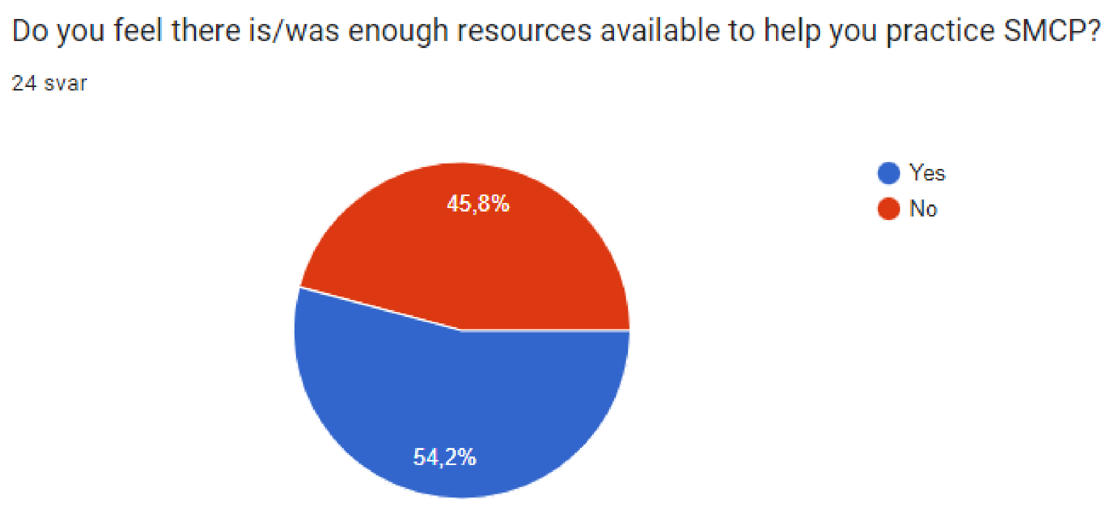
In this section, we will present the results from the method section of this project. This includes the results from the surveys, the design prototypes and the final application and code.

5.1 Data gathering

The results from the data gathering helped inform the needs of the users(students) as well as teachers teaching the subject. This helped us get a wider perspective and create a design that better aligned with these needs. In total, there were 36 participants, 28 students and 8 teachers. The responses to these questions provided valuable data and perspectives that shaped the subsequent decisions and iterations in creating an updated and user-centred application.

5.1.1 Survey results(students)

We started with a couple of general questions to get an overview of the educational landscape and see where an application for practicing SMCP would fit in.



Do you feel there is/was enough resources available to help you practice SMCP?

24 svar

Figure 1: Resource availability

Q1: If any, how do you think an application helps/helped you learn SMCP compared to traditional classroom instructions or other resources?

Answer: I think it would contribute to learning SMCP, if the user practises consistently. It would also be a more dynamic way of learning, rather than absorbing the information via passive lectures.
Answer: I've used other language learning apps like Duolingo, and I would think it would make learning SMCP a lot easier for some people.
Answer: An application would be a significant help in learning smcp because it makes learning easier to start and continue

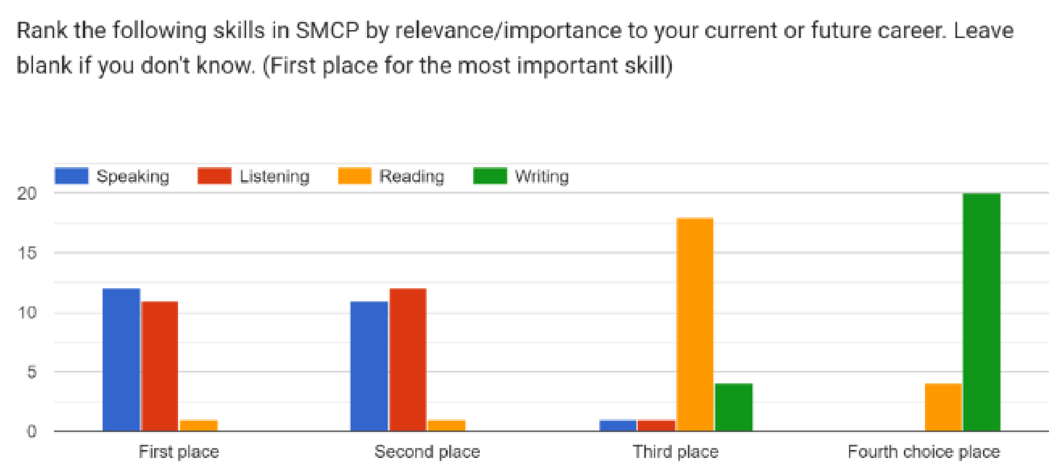


Figure 2: Skill relevance

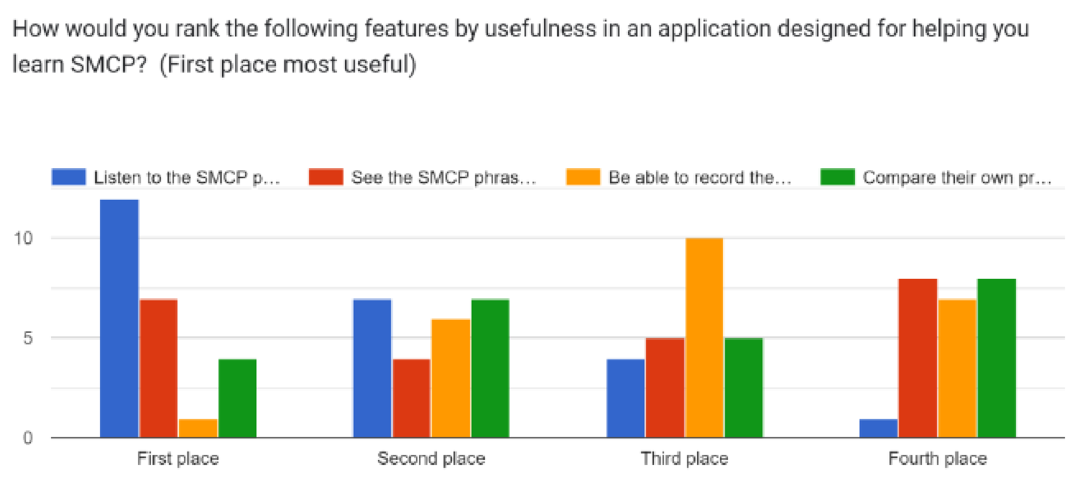


Figure 3: Feature usefulness

The feature that most students found to be the most useful/second most useful in a SMCP language learning application was to listen to the phrase. The third perceived most useful feature was for students to be able to record their own pronunciation of the phrase, followed by seeing the SMCP phrase and comparing their own pronunciation with the native speaker's in fourth place.

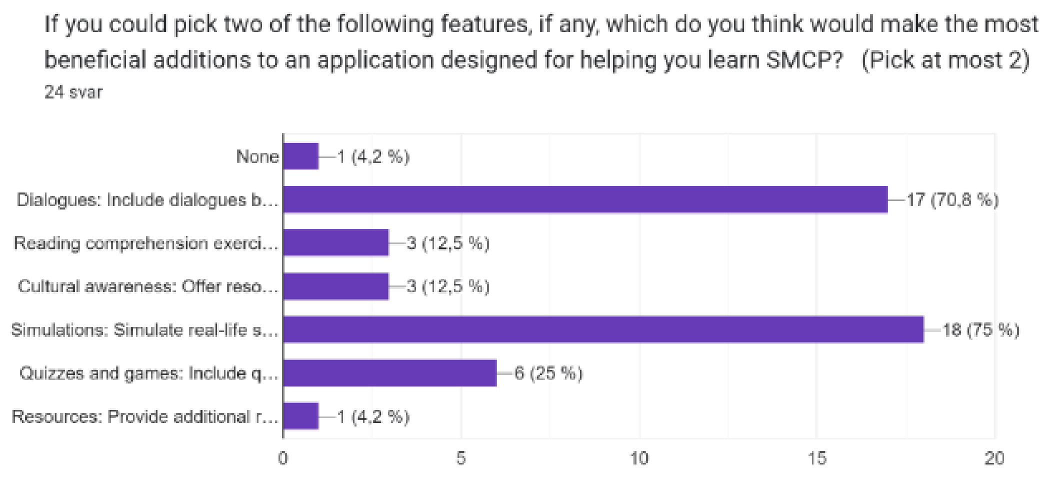


Figure 4: Additional Features(students)

Q2: Are there any other methods that you would find helpful on the user interface of a learning platform or application and if so can you give some reasons for why?

Answer: Don't have endless menus
Answer: Clear/minimalistic icons. Simple titles. No room for confusion, especially since English isn't everyone's most fluent language.
Answer: Search function in i.e. SMCP PDF document.

Q3: Do you have any additional comments or suggestions regarding a possible application dedicated to facilitating the learning of SMCP?

Answer: Include pictures when explaining certain ship specific terms.
Answer: Should be very simple interface and not require any additional knowledge/training to use. Self-explanatory.

5.1.2 Survey results(teachers)

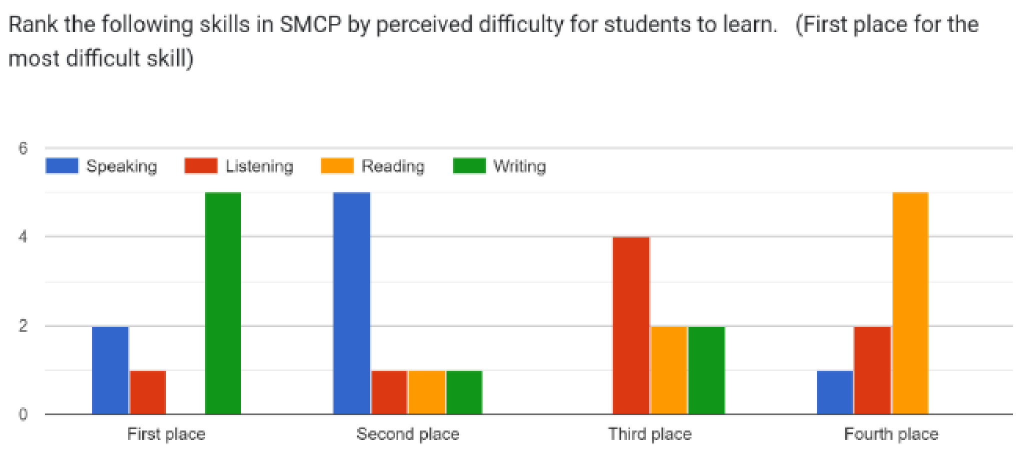


Figure 5: Skill difficulty

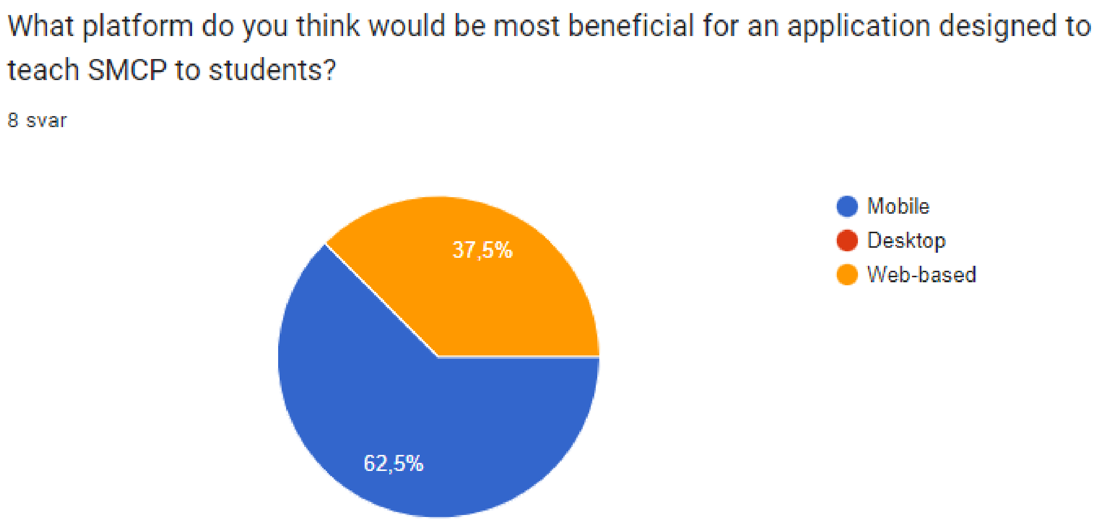


Figure 6: Suitable platform

Q4: What are the reasons for your choice?

Answer: flexibility of having your phone with you at all times
Answer: Preferably available for all platforms. Make it off-line available.
Answer: Our students are not allowed to use their mobile phones while in the Academy

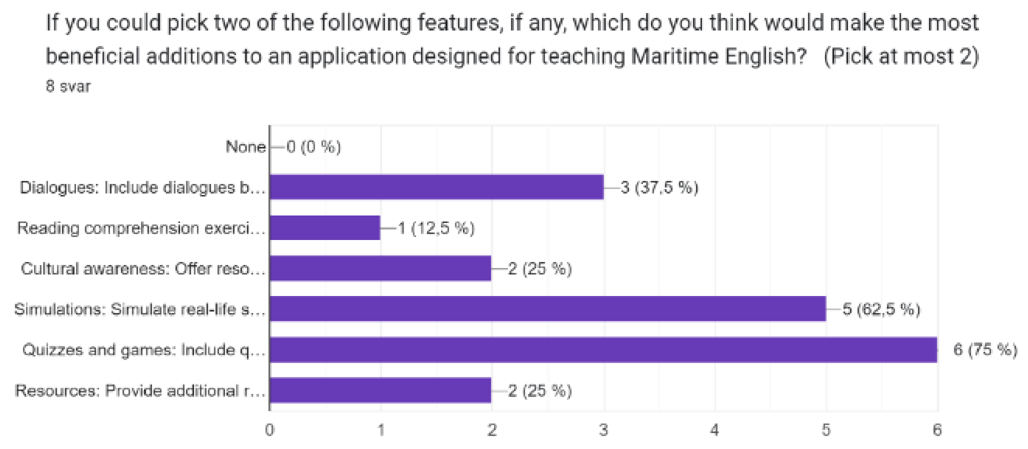


Figure 7: Additional features(teachers)

Q5: Do you have any additional comments or suggestions regarding a possible application dedicated to facilitating the learning of SMCP?

Answer: Make it user friendly and simple and free for the student to make it successful
Answer: if your application is nothing else than listen to the phrases and see them written then smcpexamples.com already exists. Do a ubiquitous tool that allows for study at a personal rate/convenience. Must be gamified too, to encourage students to use it, or else it won't.

5.1.3 Survey Analysis & Findings

Upon analyzing the survey responses from both students and teachers, several significant findings emerged, indicating similarities in their perspectives. The following key findings summarize the survey analysis:

1. **Insufficient Resources for SMCP Practice:**
   Figure 1 highlights that 45.8% of students feel that there is a lack of resources available to support their SMCP practice. This finding emphasizes the need for an assisting tool that can address this gap. Additional details from Question 1 further underscore the perceived benefits of such a tool, with respondents expressing how it would greatly enhance their practice experience.

2. **Speaking Skills and Relevance:**
   Figures 5 & 2 reveal the congruence between teachers and students regarding the challenges associated with speaking in SMCP. Teachers identified speaking as the second most difficult skill for students, while students considered it the most relevant skill for their current and future careers. This finding em-

phasizes the importance of addressing speaking skills in SMCP to meet both educational and career aspirations.

3. **Listening as a Key Feature:**
   Figure 3 indicates that students prioritize listening to SMCP phrases as the most useful feature for learning SMCP. This finding underscores the need and desire for a feature that offers ample listening opportunities. It is evident that incorporating listening exercises and resources into the application will greatly benefit students' language learning experience.

4. **Beneficial Additions:**
   In Figures 4 and 7, responses regarding beneficial additions to the application indicate that both students and teachers highly value the following:

   - Relevant dialogues (70.8% of students and 37.5% of teachers)

   - Simulations of real-life situations (75% of students and 62.5% of teachers)

   - Quizzes, and games (25% of students and 75% of teachers)

   This insight emphasizes the importance of including these features to enhance engagement and effectiveness in SMCP learning. The need for additional resources is also highlighted by teachers.

5. **User Interface Preferences:**
   Insights on the desired user interface were gained from Question 2. Students expressed the need for:
   - A search function
   - Clear and minimalistic icons
   - Simple menus.

   These preferences indicate the importance of creating a user-friendly and intuitive interface to facilitate ease of use.

6. **Platform Preference:**
   Figure 6 presents the participants' platform preferences for the application. The results indicate that 62.5% of participants chose mobile as the preferred platform, while 37.5% preferred web-based access. This indicates a clear preference for a mobile application but with a significant portion expressing interest in web accessibility as well. Thus highlighting the need to focus the application development towards a mobile platform, ideally both iOS and Android for the widest reach, while considering the possibility of supporting web accessibility to cater to an even wider audience.

7. **Additional Comments and Requests:**
   Question 3 revealed specific requests, such as the desire for a potential ship game to aid in learning the different parts of a ship. The need for a potential

dictionary was highlighted in question 2, where a user preferred a search function in the IMO SMCP document. Additionally, the need for simplicity and ease of use was emphasized by teachers in Question 5.

These survey findings provided valuable insights into the needs, preferences, and suggestions of both students and teachers. Incorporating these findings into the application's design and development process helped ensure that it effectively addressed the identified needs and aligned with user expectations, ultimately enhancing the SMCP learning experience.

5.2 Prototyping

The result from the design prototype consisted of a hand-drawn Wireframe sketch and a high-fidelity Figma design. The Wireframe outlined the skeleton of the application while the Figma design depicted a more realistic application with colors and user interaction capability.

5.2.1 Wireframe

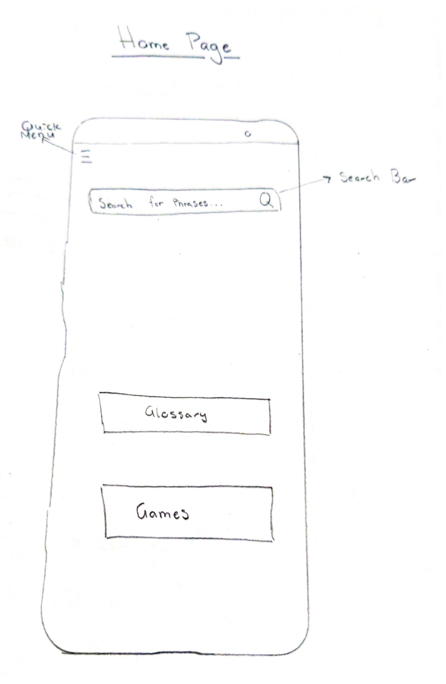


Figure 8: Wireframe home page

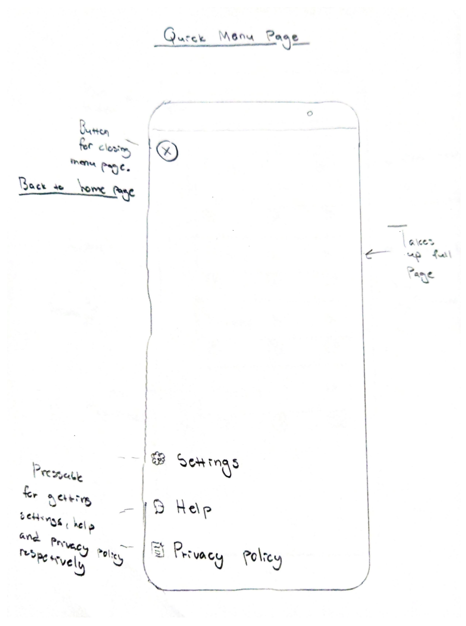

Quick Menu Page
Button for closing menu page.
Back to home page
Takes up full Page
Settings
Help
Privacy policy
Pressable for getting settings, help and privacy policy respectively

Figure 9: Wireframe quick menu

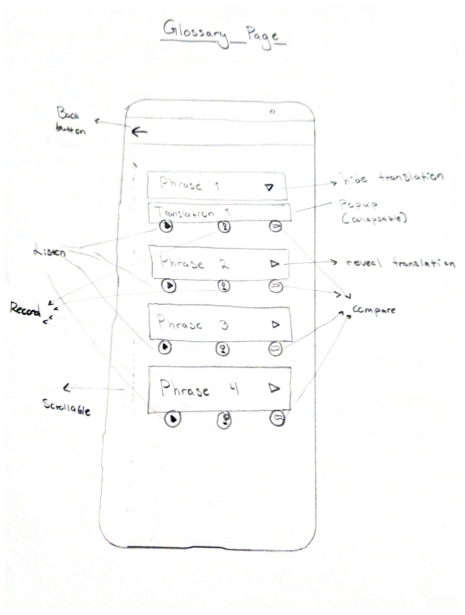


Figure 10: Wireframe practice page

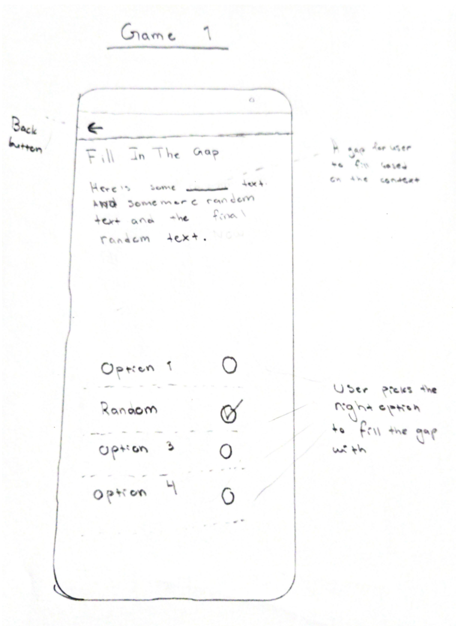

Game 1
Back button
Fill In The Gap
Here is some ____ text and some more random text and the final random text.
A gap for user to fill based on the context
Option 1
Random
option 3
option 4
User picks the right option to fill the gap with

Figure 11: Wireframe missing word game

The home page (see Figure 8) is the initial page a user will be presented with upon launching the application. The page is designed to allow for easy navigation. It includes features like accessing a quick menu, a search bar for searching for phrases and navigating to the glossary or games.

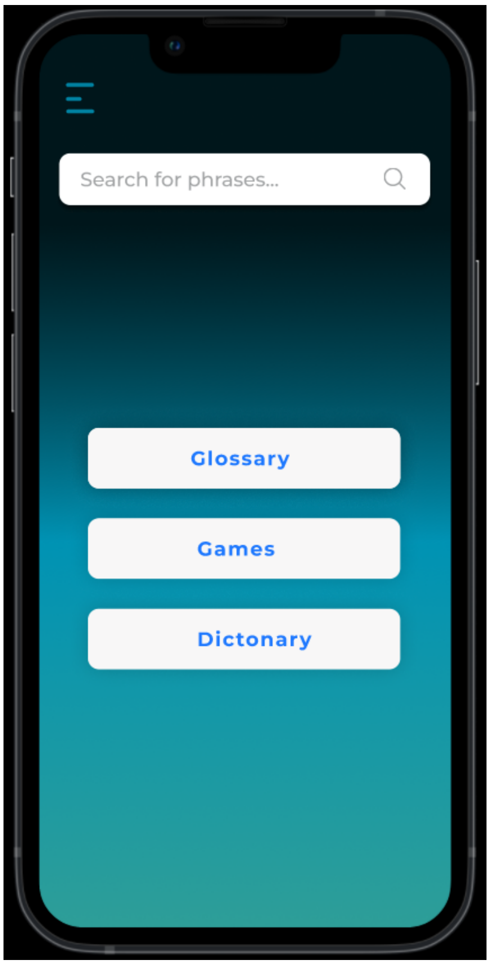
The quick menu page (see Figure 9) can be accessed from the home page (see Figure 8) and navigates back there. The page is designed to allow the user for quick access to supplemental features that helps the user make better use of the application. From this page, a user can access settings, help and privacy policy.

The glossary practice page (see Figure 10) can be accessed from the home page (see Figure 8) and navigates back there. The page supports the following features:

- The user can play a phrase to hear the correct pronunciation
- Record themselves pronouncing the phrase
- Compare their own recording with the correct pronunciation
- Reveal/unreveal a translation of the phrase in their local language if supported.

The missing word game page (see Figure 11) can be accessed from the home page (see Figure 8) and navigates back there. The page is designed to host a mini-game that allows the user to practice their phrases in a more contextual way. The user is provided with a text with a word missing and is given 4 options to fill the gap, with one of the options being the correct missing word.

5.2.2 Figma prototype



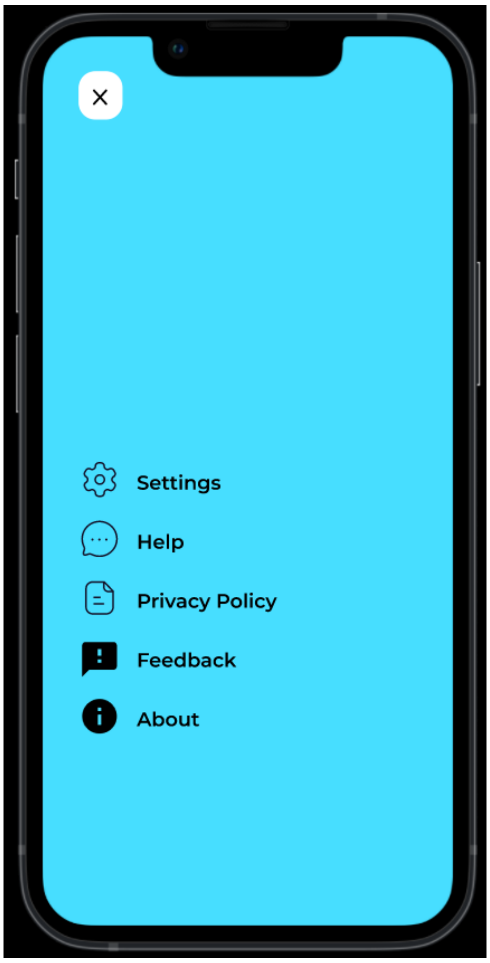


Figure 12: Figma home page

Figure 13: Figma quick menu page

The home page (see Figure 12) was implemented according to the wireframe design (see Figure 8). An additional selection for a dictionary feature was added. The quick menu page (see Figure 13) was implemented according to the wireframe design (see Figure 9). Additional selections for feedback and an about section were added.

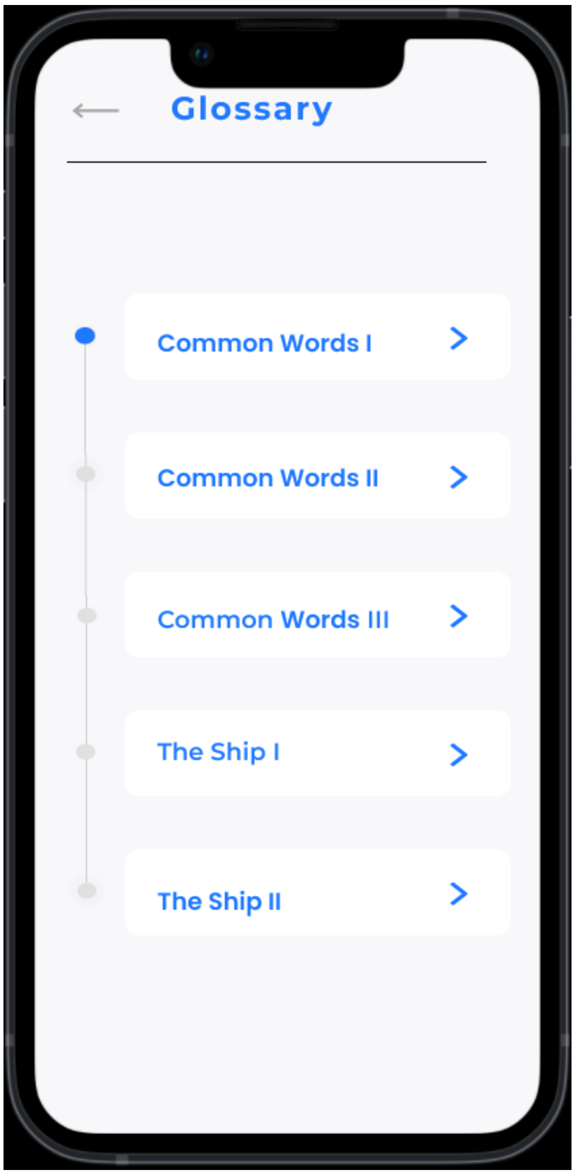


Figure 14: Figma glossary menu page

The glossary menu page (see Figure 14) can be accessed from the home page (see Figure 12) and navigates back there. The page is designed to allow the user for easier and quicker access to different glossary sections (See Figure 15) and help divide the learning into different compartmentalized sections. The page also supports progress tracking on the different sections with completed ones showing up with a blue dot on their left side and uncompleted ones being the unfilled dot.

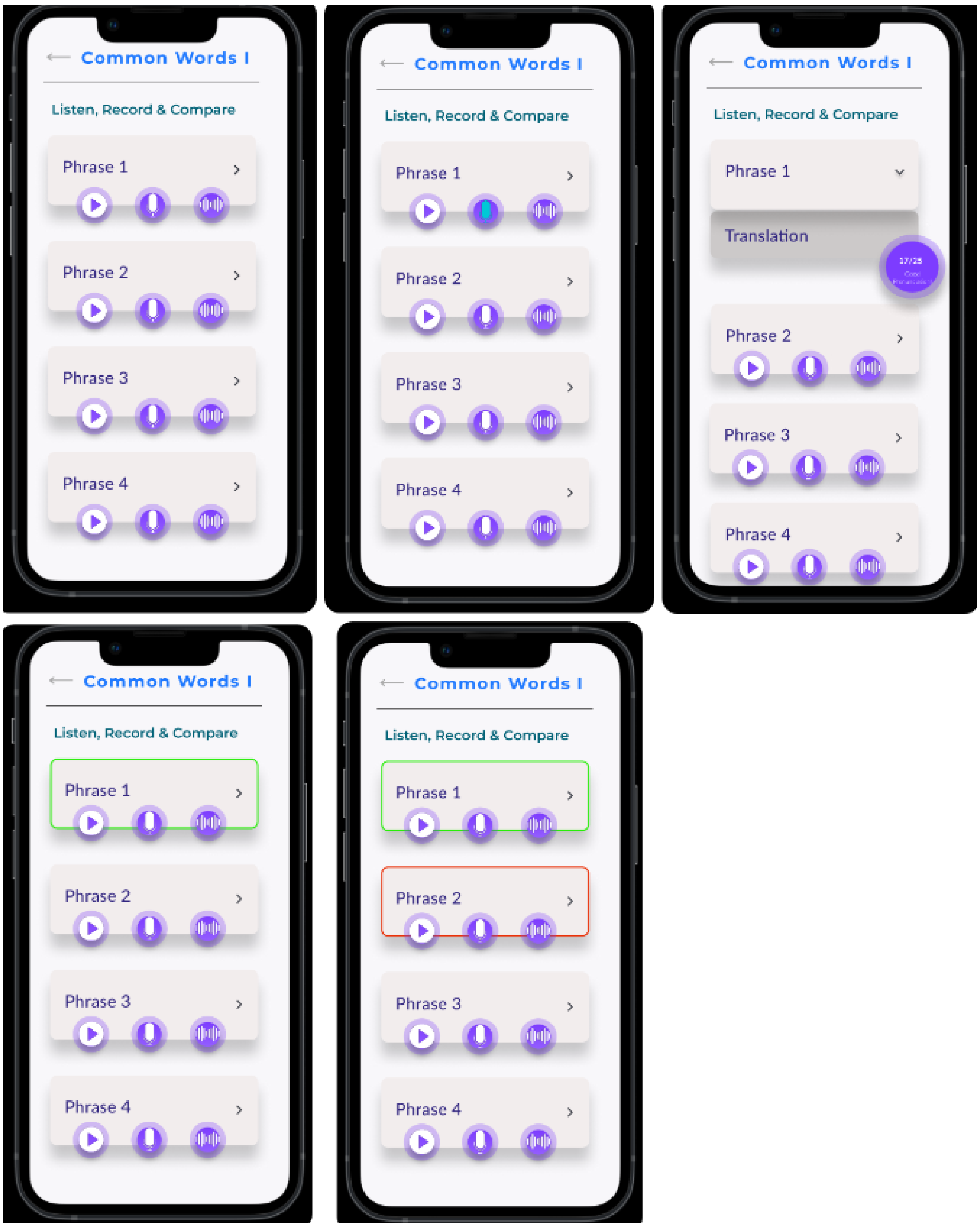


Figure 15: Figma glossary practice page

The glossary practice page (see Figure 15) was implemented according to the Wireframe design (see Figure 10). Additional features were added such as revealing a score on pronunciation correctness, highlighting a correctly completed phrase in green and an incorrectly completed one in red. A header text instructed the user to 'Listen, record & compare' which served as an additional hint to the way the buttons were ordered.

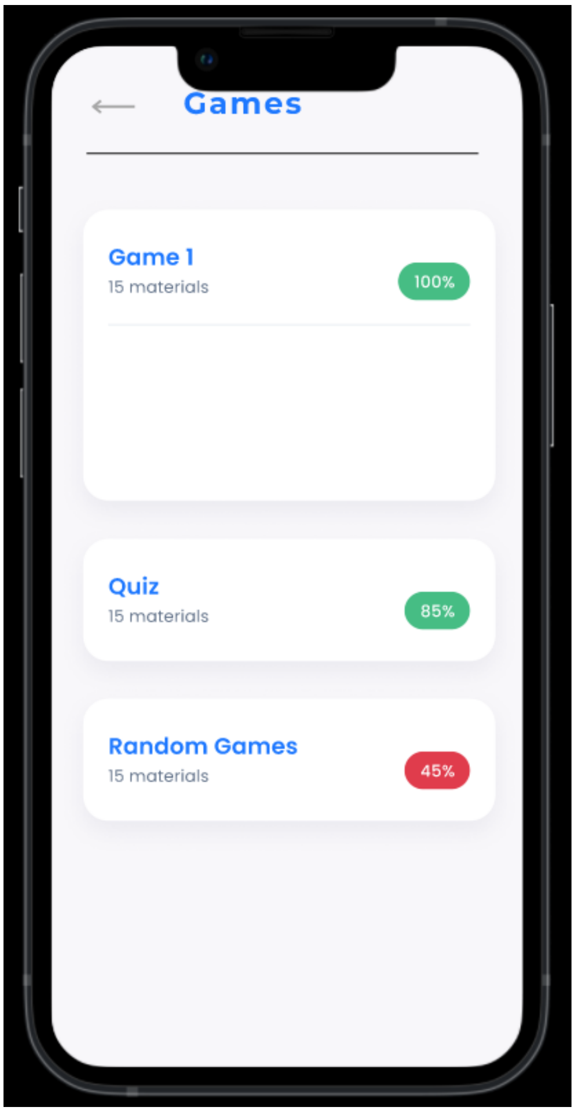


Figure 16: Figma game menu page

The games menu page (see Figure 16) can be accessed from the home page (see Figure 12) and navigates back there. The page is designed to allow the user for easier and quicker access to different supported games (See Figure 17) and help divide the learning into different compartmentalized sections. The page also supports progress tracking on the different games with the rate of completion showing up in both percentage and colour coding, red meaning considerably in the uncompleted range and green considerably in the completed range.

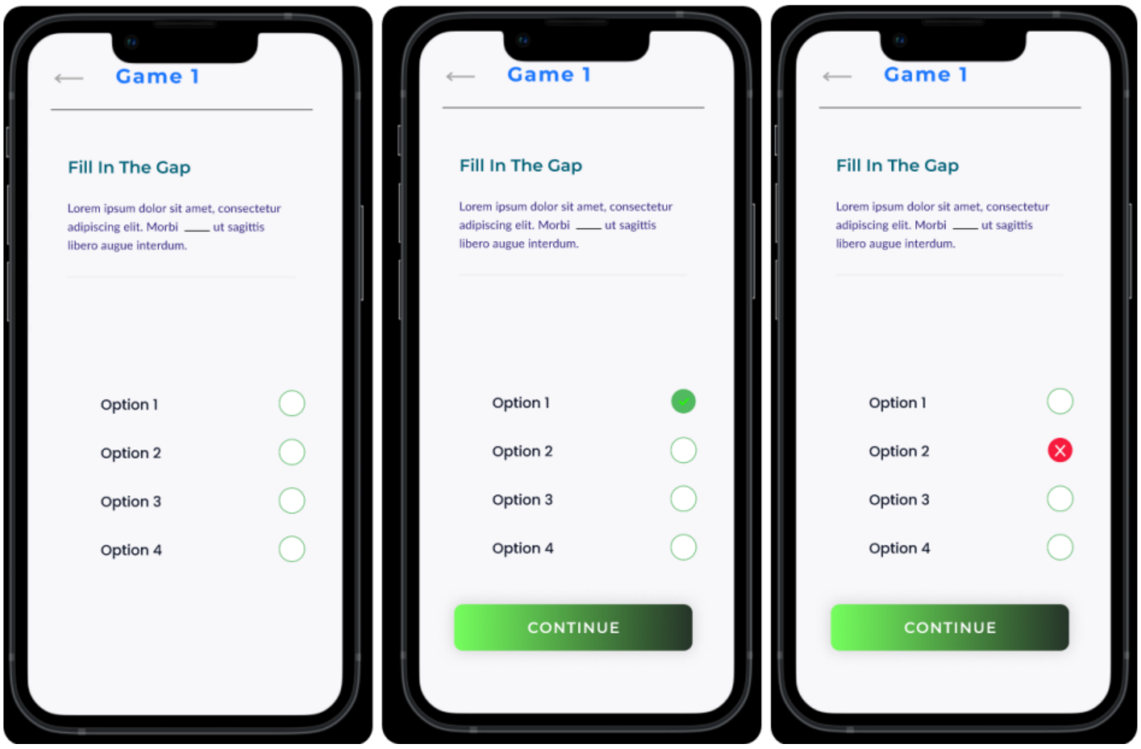

Game 1
Fill In The Gap
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi ___ ut sagittis libero augue interdum.
Option 1
Option 2
Option 3
Option 4
CONTINUE

Figure 17: Figma missing word game page

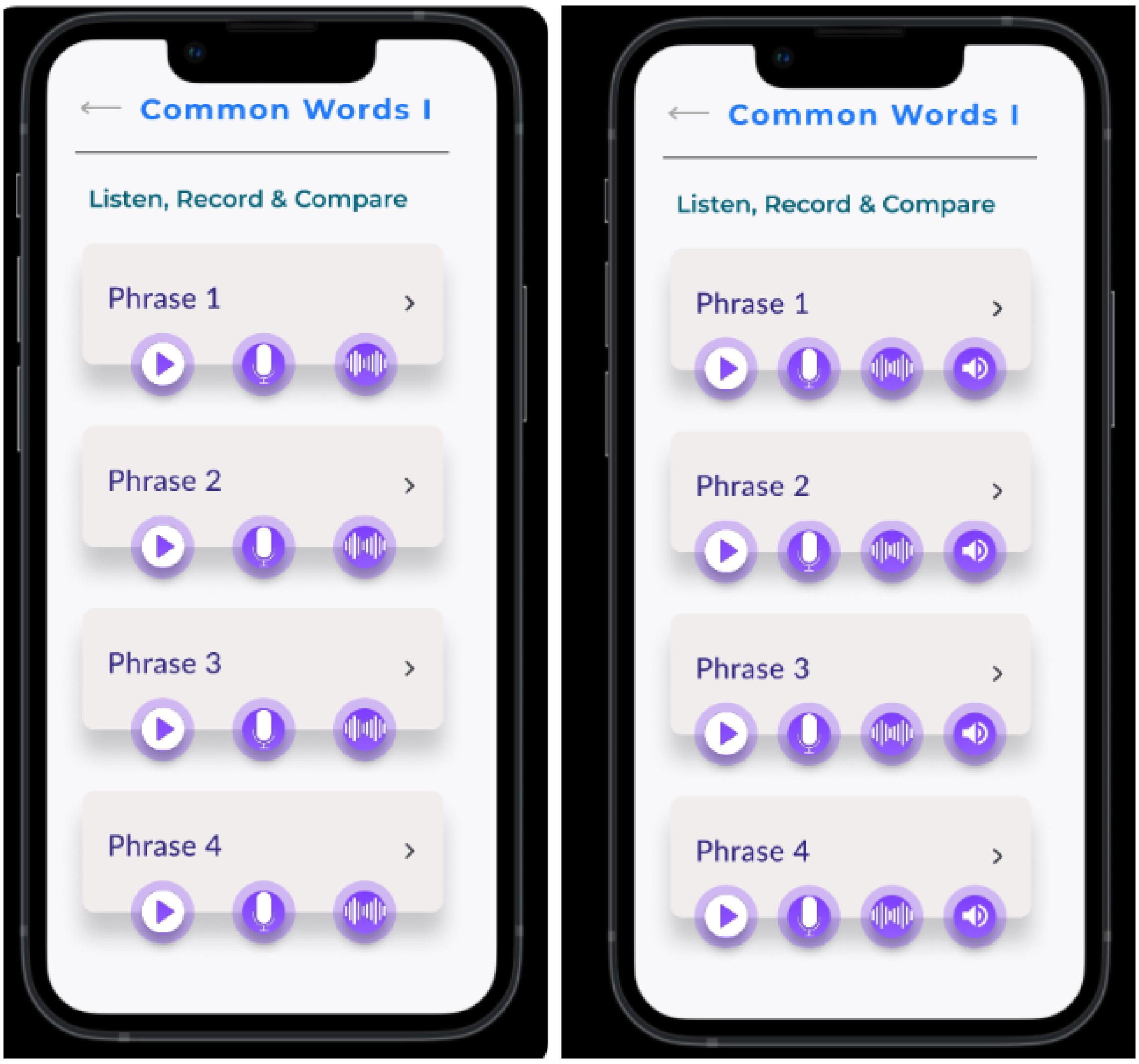
The missing word game page (see Figure 17) was implemented according to the Wireframe design (see Figure 11). Added additional features such as highlighting a correct or incorrect answer, correct being a green check and incorrect a red cross. A continue button appears after a guess to navigate to the next iteration of the game.

5.3 Evaluation

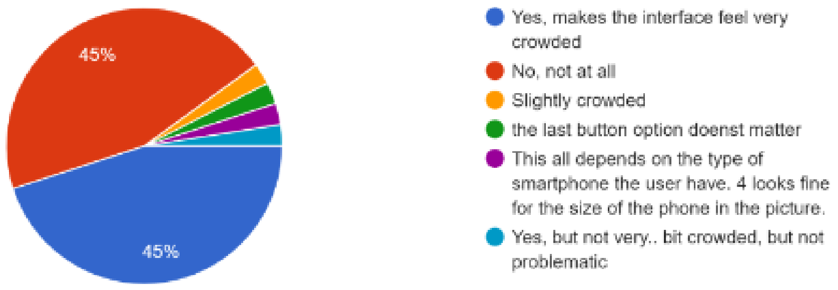
The results from the evaluation phase consisted of the results from the evaluation survey that was sent out to students internationally. This survey evaluated the Figma prototype. However, since we had also started building the app at this point, we also evaluated the design of our actual Flutter application to gain valuable insights from the users.

5.3.1 Survey results(students)

The responses to these questions provided valuable data and perspectives that shaped the subsequent decisions and iterations in creating an updated and user-centred prototype. In total, there were 40 participants, all students. By targeting students specifically, the survey ensured that the perspectives of the primary users of the application were captured comprehensively.

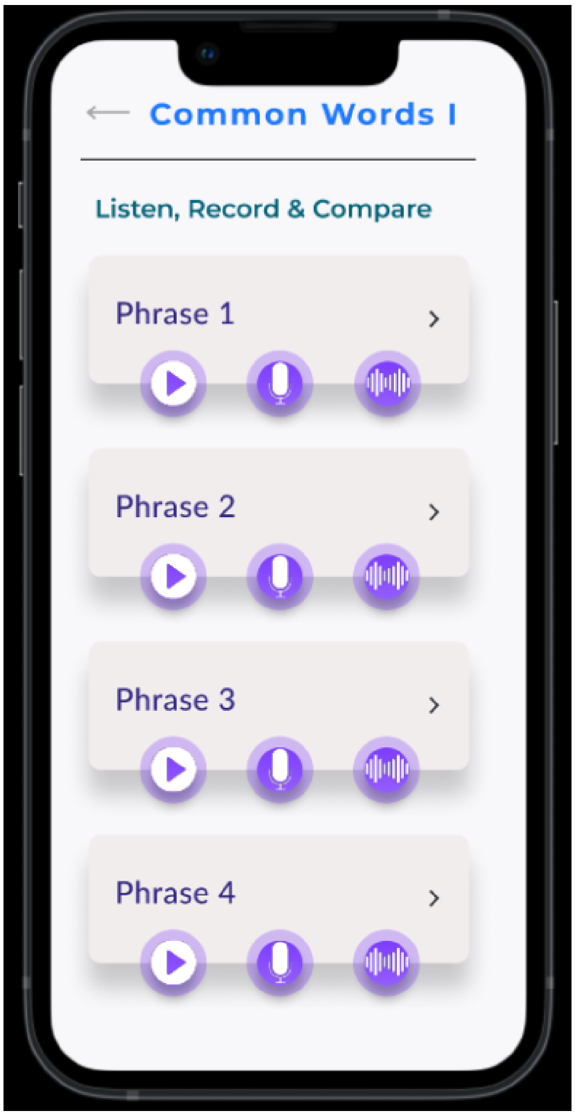


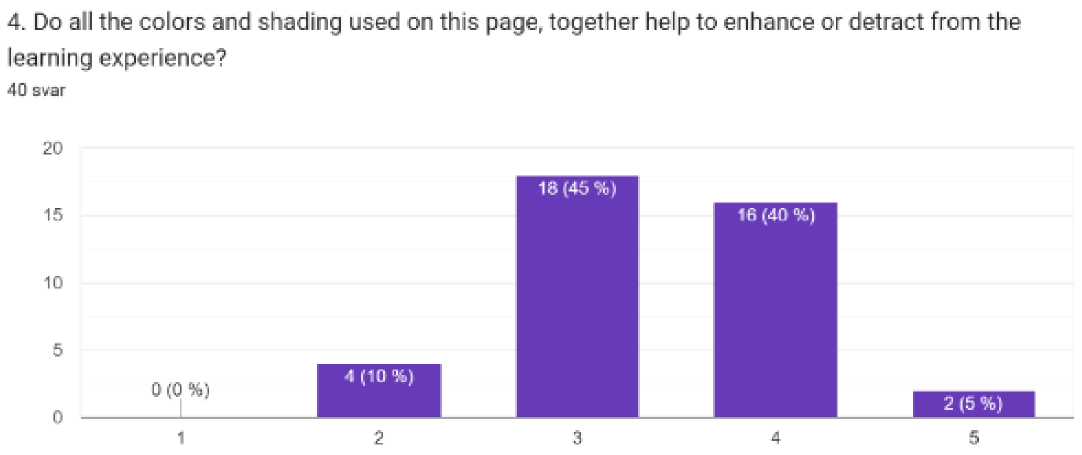
3. Does having four icons per row make the interface feel crowded? (For the page design above)
40 svar



- ● Yes, makes the interface feel very crowded
- ● No, not at all
- ● Slightly crowded
- ● the last button option doenst matter
- ● This all depends on the type of smartphone the user have. 4 looks fine for the size of the phone in the picture.
- ● Yes, but not very.. bit crowded, but not problematic

Figure 18: Number of icons




4. Do all the colors and shading used on this page, together help to enhance or detract from the learning experience?

40 svar

Figure 19: Colors and shading

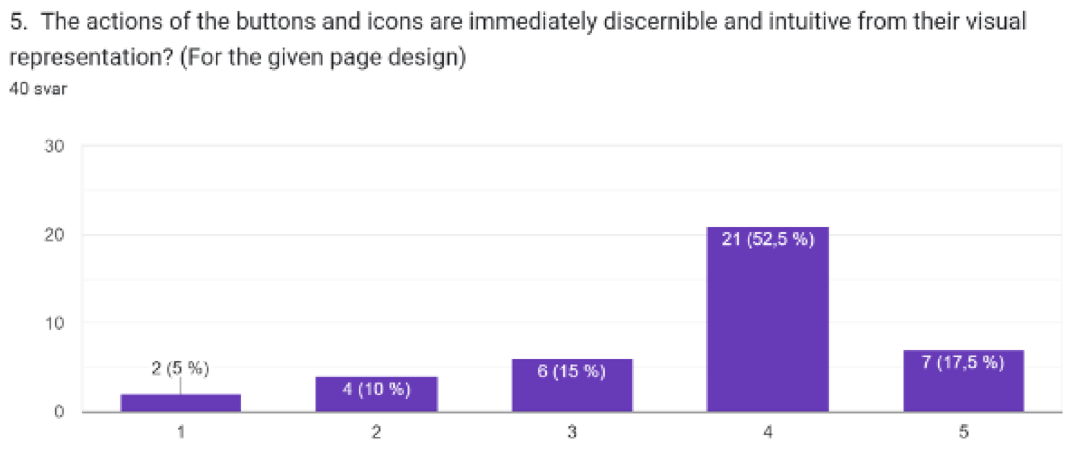
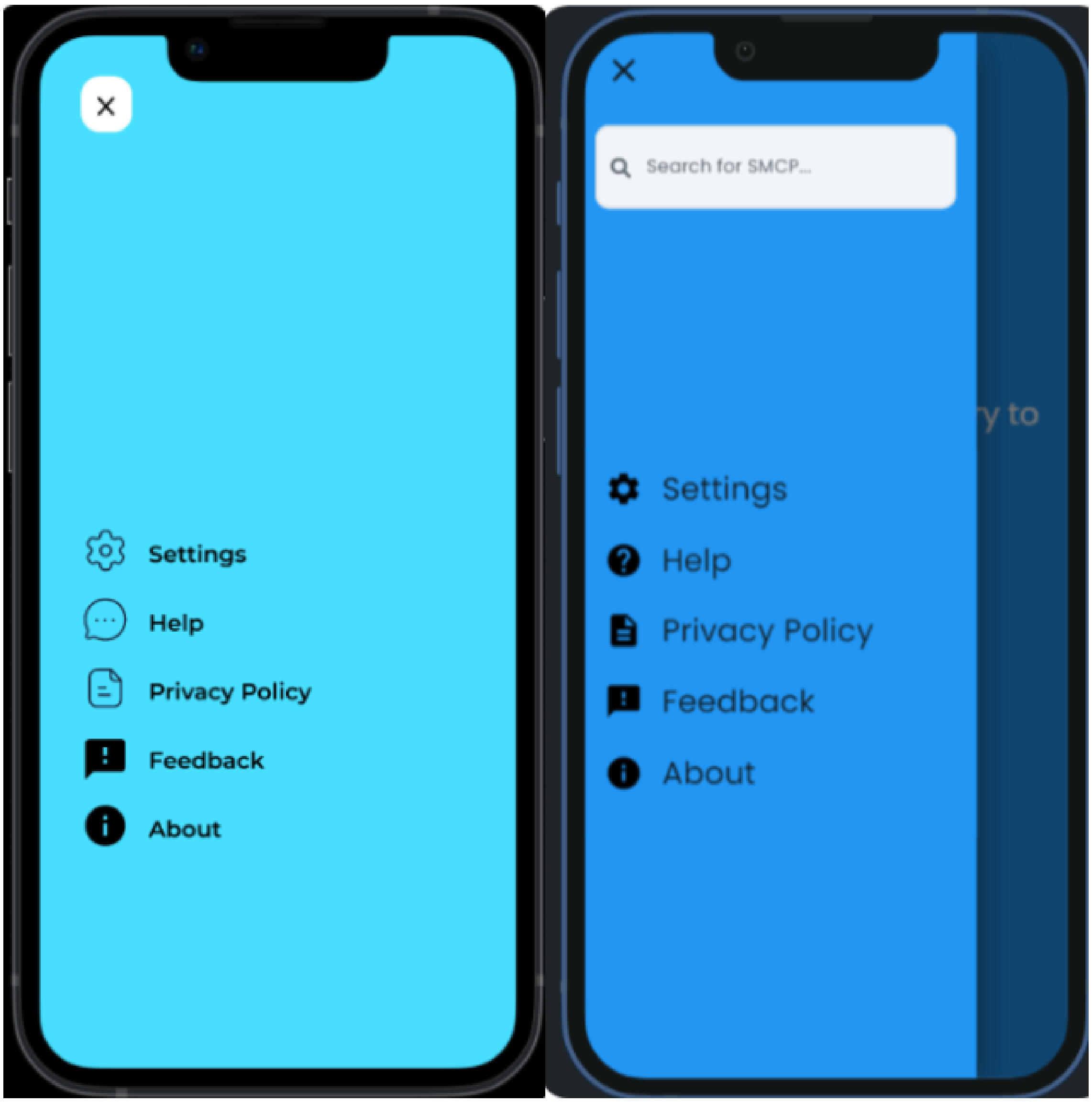


Figure 20: Button and icons intuitiveness

Q6: If you disagree please explain why
Answer: The difference between the 2nd and 3rd button is not really clear
Answer: Seems weird with 3 different colors for this screen. The blue, green and purple, or if there's some intention behind this.
Answer: What is the difference between play and the soundwaves one?
Answer: I am not too sure about the compare Icon.
Answer: I cannot understand what those 3 buttons are for which one plays the sentence why is there a microphone
Answer: I do not understand the third one



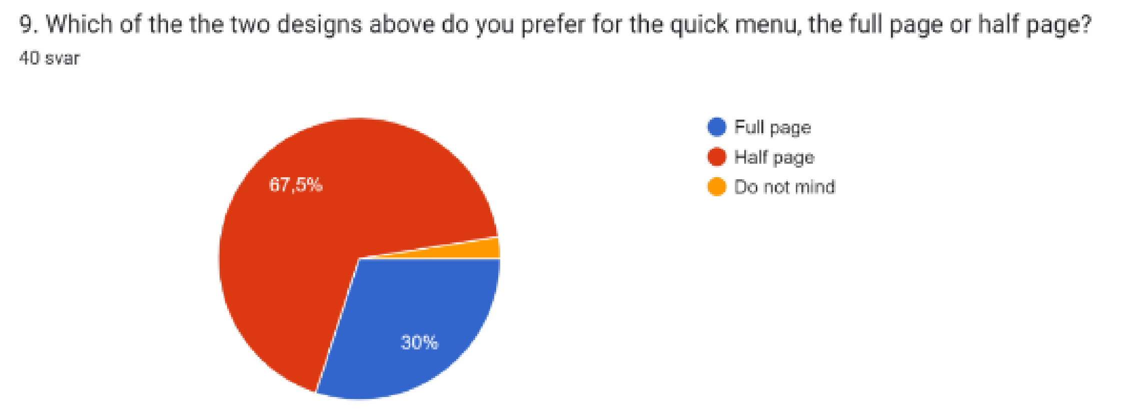
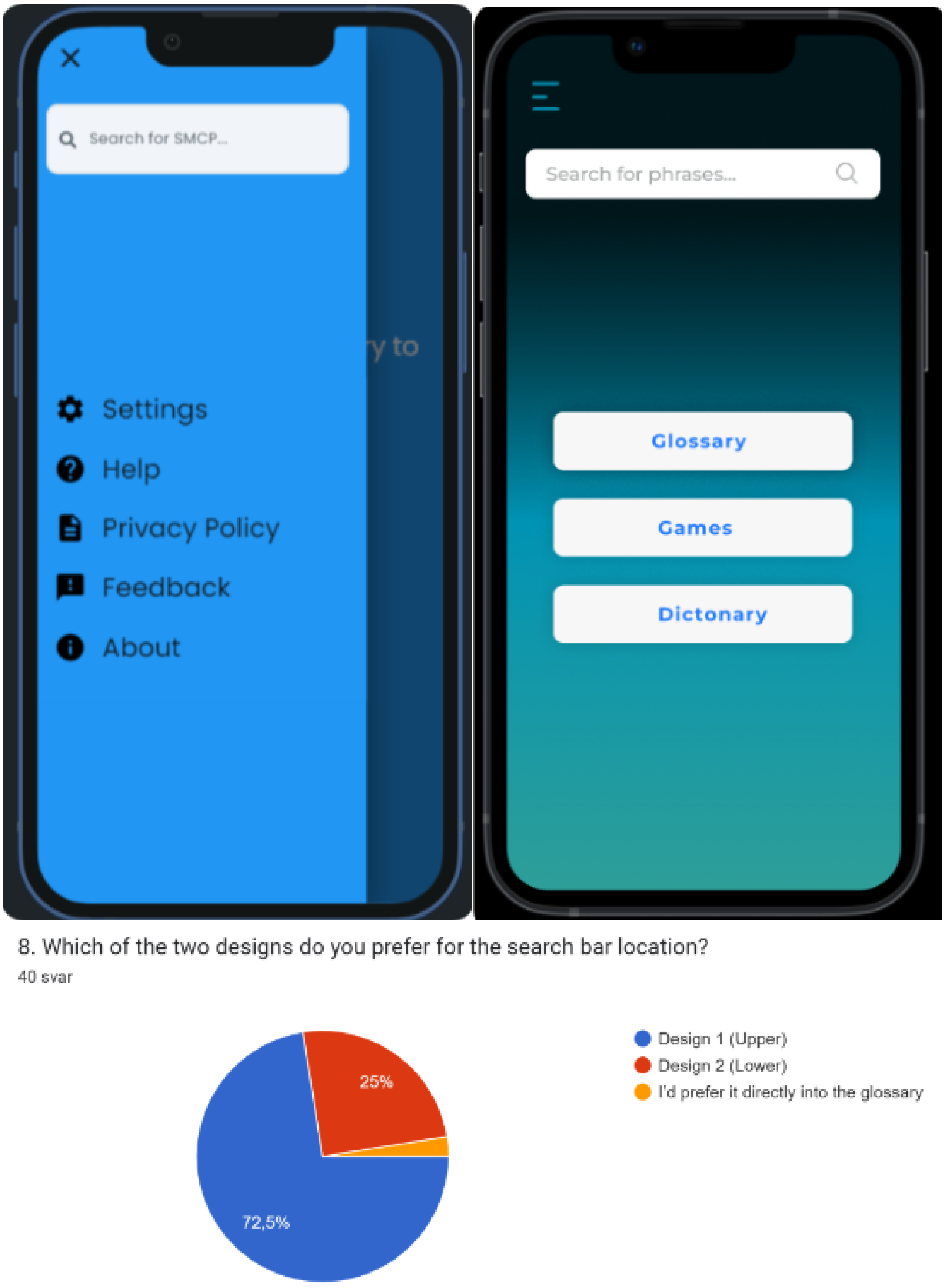

9. Which of the the two designs above do you prefer for the quick menu, the full page or half page?
40 svar

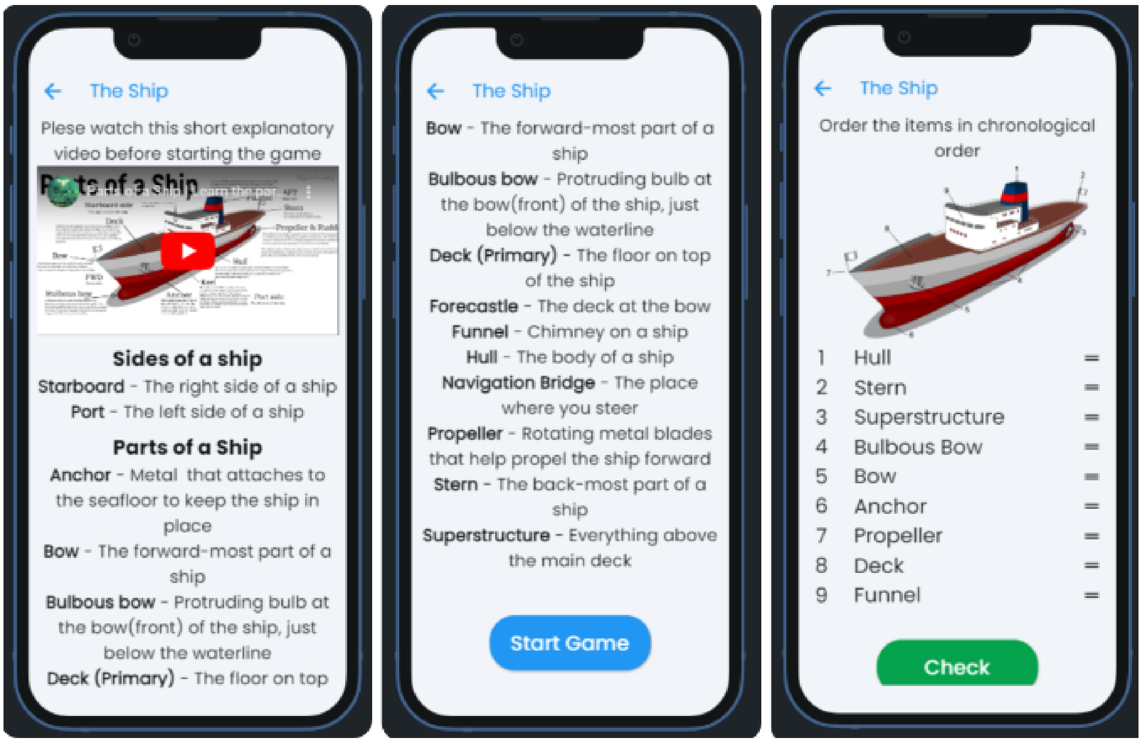
Figure 21: Full page or half page



8. Which of the two designs do you prefer for the search bar location?
40 svar

Figure 22: Search bar location

*Design 1 (upper) = Home page search bar
*Design 2 (lower) = Quick Menu search bar



Plese watch this short explanatory video before starting the game

Sides of a ship
Starboard - The right side of a ship
Port - The left side of a ship

Parts of a Ship
Anchor - Metal that attaches to the seafloor to keep the ship in place
Bow - The forward-most part of a ship
Bulbous bow - Protruding bulb at the bow(front) of the ship, just below the waterline
Deck (Primary) - The floor on top

Bow - The forward-most part of a ship
Bulbous bow - Protruding bulb at the bow(front) of the ship, just below the waterline
Deck (Primary) - The floor on top of the ship
Forecastle - The deck at the bow
Funnel - Chimney on a ship
Hull - The body of a ship
Navigation Bridge - The place where you steer
Propeller - Rotating metal blades that help propel the ship forward
Stern - The back-most part of a ship
Superstructure - Everything above the main deck

Order the items in chronological order

1 Hull
2 Stern
3 Superstructure
4 Bulbous Bow
5 Bow
6 Anchor
7 Propeller
8 Deck
9 Funnel

10. Is the ship page too crowded? (left and middle images)
40 svar

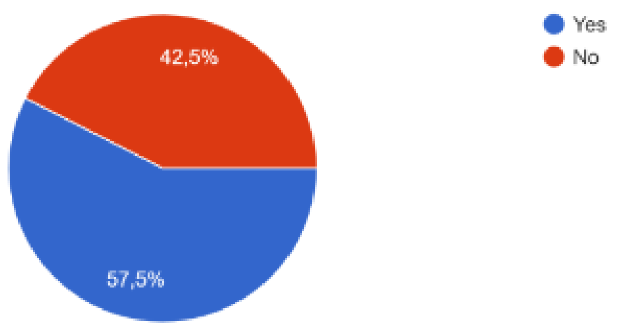
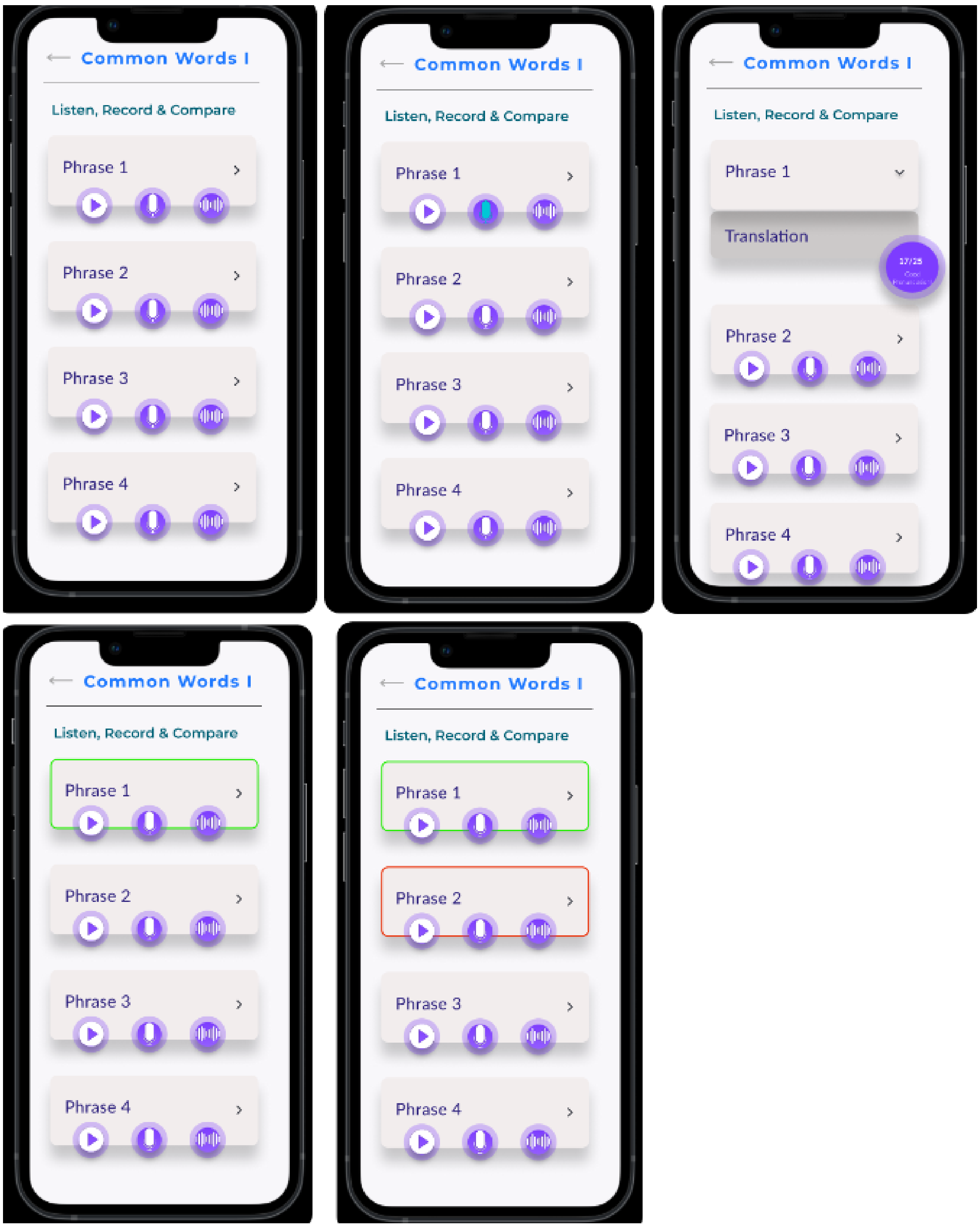


Figure 23: The Ship Page user friendliness

Q7: If yes please give some explanation for why and what could be removed
Answer: Maybe a drop-down menu when pressing on “sides” and ”parts” etc.
Answer: Drop window for ship specs
Answer: Pictures 1 and 2 are crowded partly because you chose to center the text. Best option would be to hide the definition at first and let the user click to open the text for each item
Answer: there is too much text i think it’s not really attractive to look at



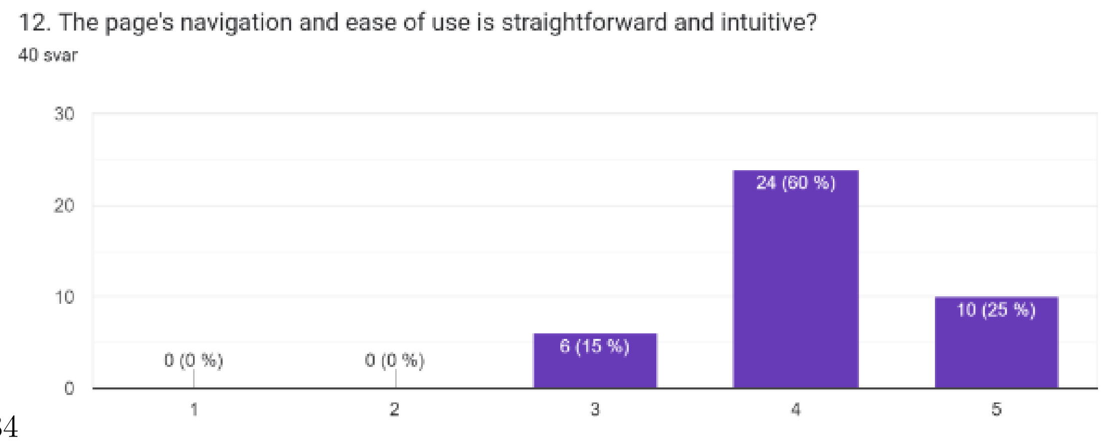


Figure 24: Page navigation and ease of use

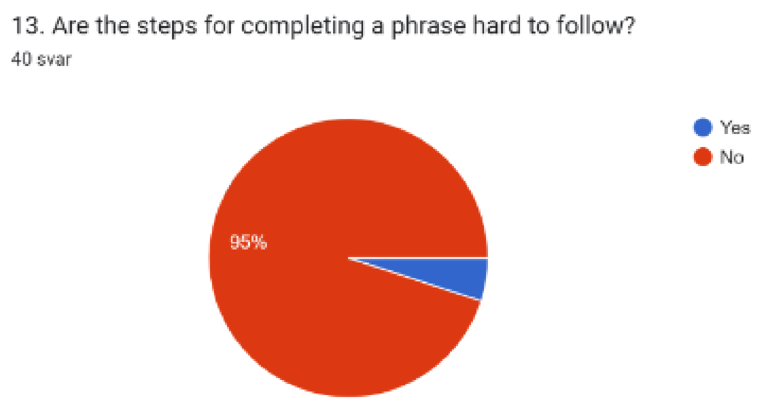


Figure 25: Phrase completion steps difficulty

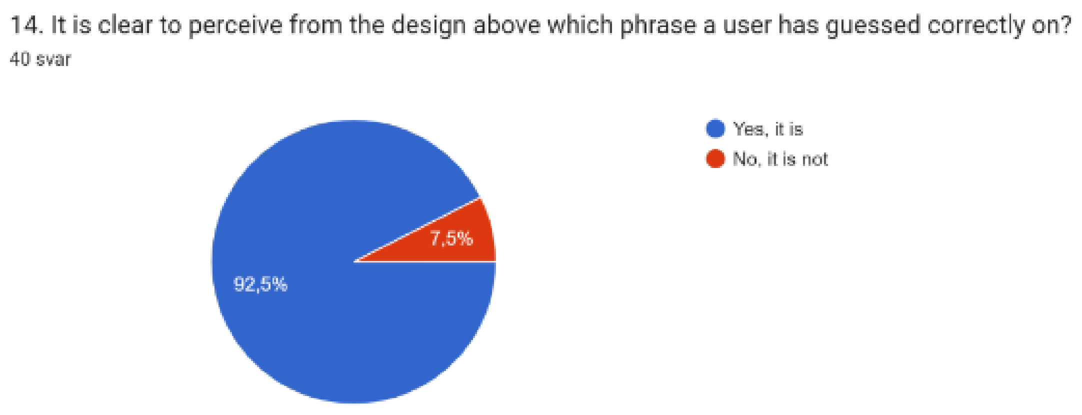


Figure 26: Perceiving completion of phrase difficulty

Q8: If you answered no please explain why?
Answer: Symbols are better than color identification.

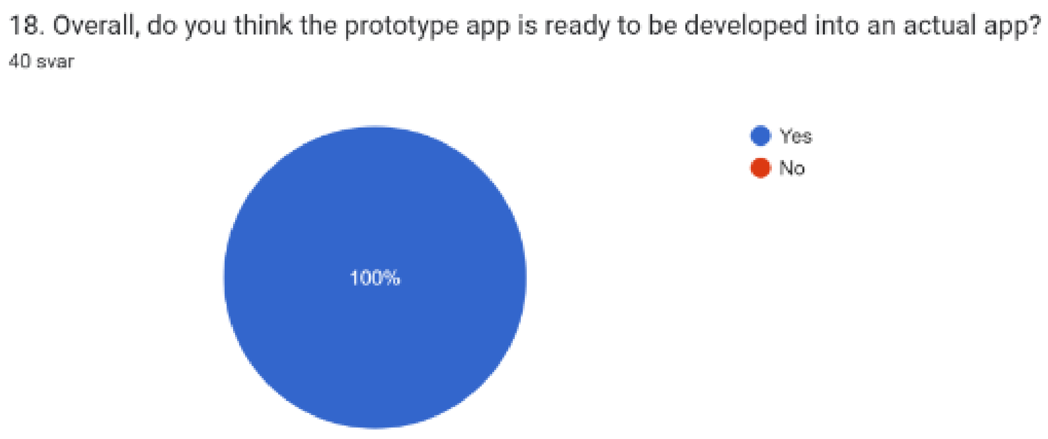


Figure 27: Prototype readiness

Q9: Can you please motivate your answer
Answer: there is few little changes to do but the app could be used and reach its purpose without problem.
Answer: I think that the prototype already looks pretty good and easy to navigate
Answer: It seems very clear and already useful to students to practice their SMCP, with different possibilities to practice (glossary, games, dictionary)

5.3.2 Survey Analysis and Findings

After collecting the survey responses, a thorough analysis was conducted to extract key findings and identify patterns in the data. The survey analysis provided valuable insights into the strengths, weaknesses, and areas for improvement in the prototype as well as the app. The following key findings summarize the survey analysis:

Design:

- **Font and Text Size:** Participants indicated that the chosen font and text size was to their liking.
- **Colors and Shading:** As can be seen in Figure 19, the majority of participants found that the colours and shading employed in the design neither enhanced nor detracted from the learning experience. However, a large number of participants thought it helped to enhance the learning experience. One person commented that using 3 different colors in the same screen seemed a bit off.
- **Buttons and Icons:** Figure 20 showed that majority of participants found the actions of the buttons and icons immediately discernible and intuitive from their visual representation. However, a larger minority did not, where several people did not understand the purpose of the 3rd button. This leaves room for improvement and incentive for a help page that explains the function of the buttons.

An important design aspect learned was the number of icons the Glossary Section page should have per row. As additional icons were potentially being considered, Figure 18 indicated that 45% of participants believed that four icons or more would be too overcrowded for the interface, suggesting that four icons or more wouldn't have clear majority support.

Layout:

- **Quick Menu Page Layout:** As can be seen in Figure 21, 67.5% of participants preferred the half-page look for the quick menu page layout. Indicating that implementing it instead of the current full-page look might be a viable choice.
- **Search Bar Location:** Figure 18 showed that majority of participants(72.5%) preferred to have the search bar in the home page instead of in the quick menu.
- **Ship Game layout:** Figure 19 indicates that the layout of the ship game is too crowded and needs to be adjusted. The need for a collapsable/scrollable text is further highlighted in question 7.

Workflow:

- **User-Friendly and Intuitive:** Figure 24 shows that participants found the prototype application to be straightforward and intuitive to use.
- **Clear Identification of Incorrect Guesses:** Figure 26 indicates that the majority of participants found it clear which phrase a user has guessed incorrectly on. However, 7.5% of participants did not find it clear, and one participant mentioned in response to Q8 that "Symbols are better than colour

identification". This aspect presents an opportunity for improvement.

Overall:

- **Prototype Readiness:** As can be seen in Figure 27, there was a clear majority of 100% who believed that the prototype was ready to be developed into an actual app.
- **Positive Feedback:** When asked to motivate in Q9, participants commonly mentioned that the prototype was clear to use and looked easy to navigate. Many participants also found the extra additions of games to be helpful.

These survey findings, as highlighted in the respective figures, provided valuable insights into the strengths, weaknesses, and areas for improvement in the prototype. Incorporating these findings into the iterative design and development process helped refine the prototype as well as the application and ensure it aligned with user preferences and expectations.

5.4 Software implementation

The results from the software implementation consisted of a fully functional cross platform application built using the Flutter framework and the Dart programming language.

5.4.1 Flutterflow

The majority of the UI was built using Flutterflow's visual drag and drop builder. Figure 28 shows the Flutterflow workspace.

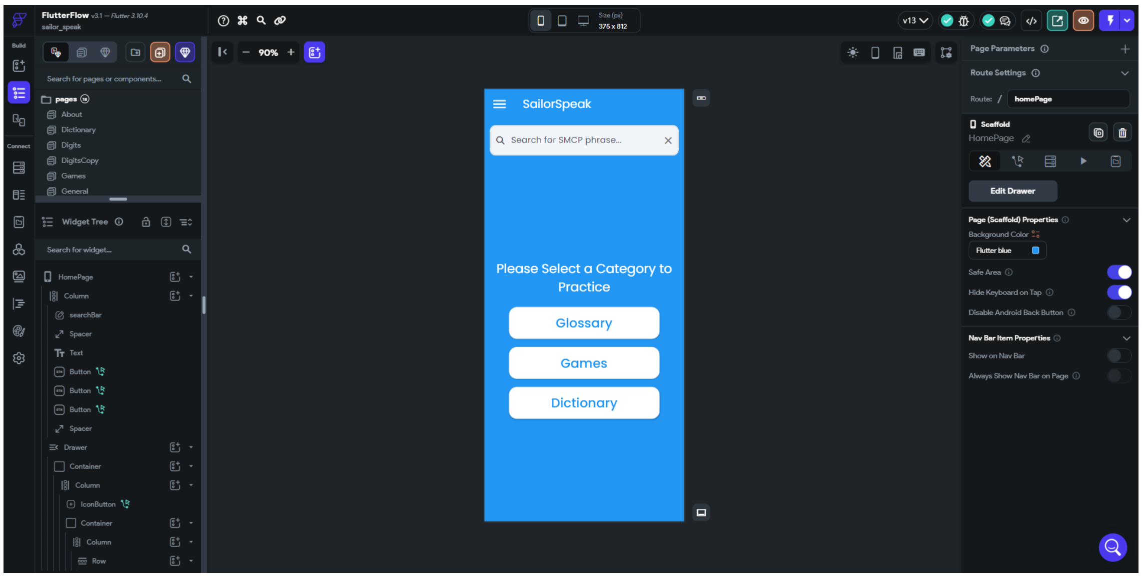


Figure 28: Homepage in Flutterflow workspace

To the left are all of the pages in the application as well as all of the components that make up those pages, such as buttons, text widgets, TextField widgets etc. To the right is the option to customize the widgets, such as changing the size and text of the buttons, changing the color of the text and other functionality associated with the specific widget.

When starting the application, the user would be presented with a splash screen.

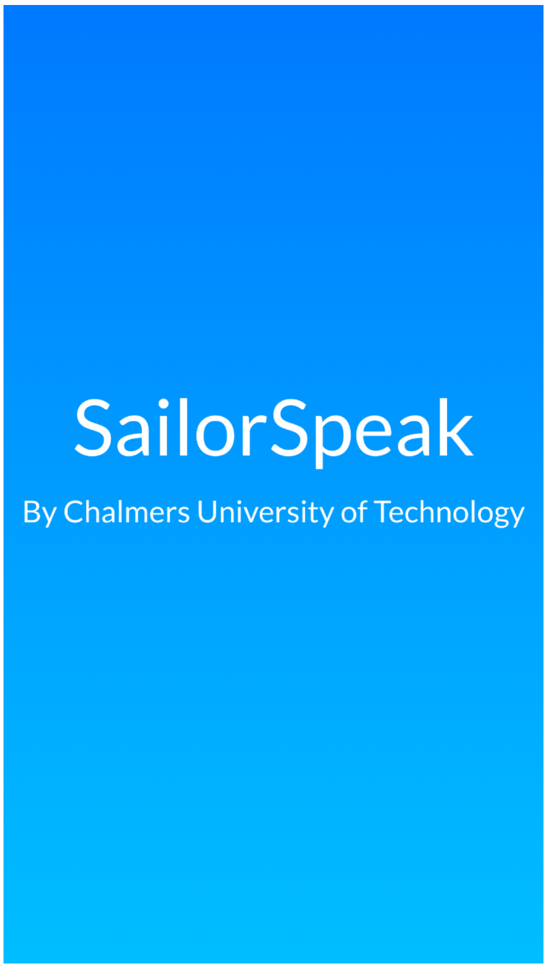


Figure 29: Splash Screen presented to user when starting the application

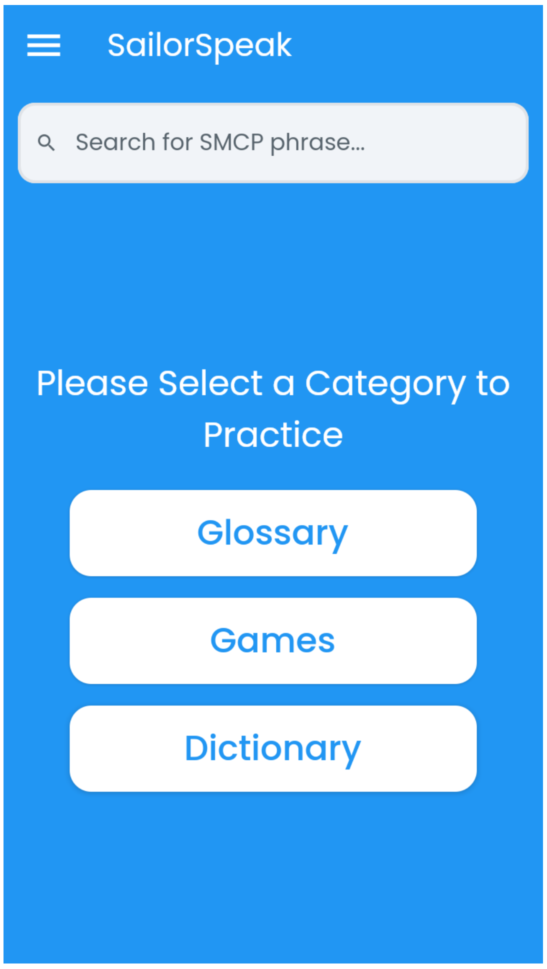

SailorSpeak
Search for SMCP phrase...
Please Select a Category to Practice
Glossary
Games
Dictionary

Figure 30: Home page created in Flutter

The home page consisted of 3 main sections, a glossary section, a games section and a dictionary, as well as a home menu.

5.4.2 Home menu

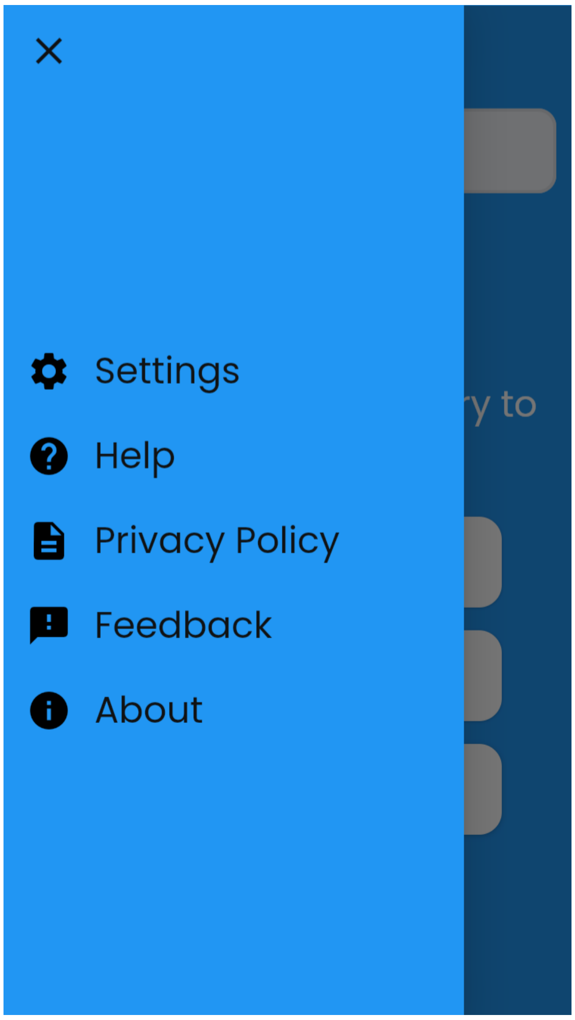


Figure 31: Home menu in the Flutter application

The home menu consisted of 5 items:

- **Settings:** This page controlled the settings of the application. The only setting we implemented was light-/dark mode.
- **Help:** This page contained guides on how to use the different parts of the application.
- **Privacy Policy:** This page was aimed for the privacy policy. However, since our application did not collect any user sensitive data, we did not have a privacy policy.
- **Feedback:** The purpose of this page was to link the users to a feedback form where they could fill out their experience of the final application. Similarly to how we had evaluated the design(see 5.3), we also wanted to evaluate the final application. However, since we finished the application later than we expected, we did not carry through with that idea.
- **About:** This section was devoted to describing the purpose of the application and why/by whom it was created.

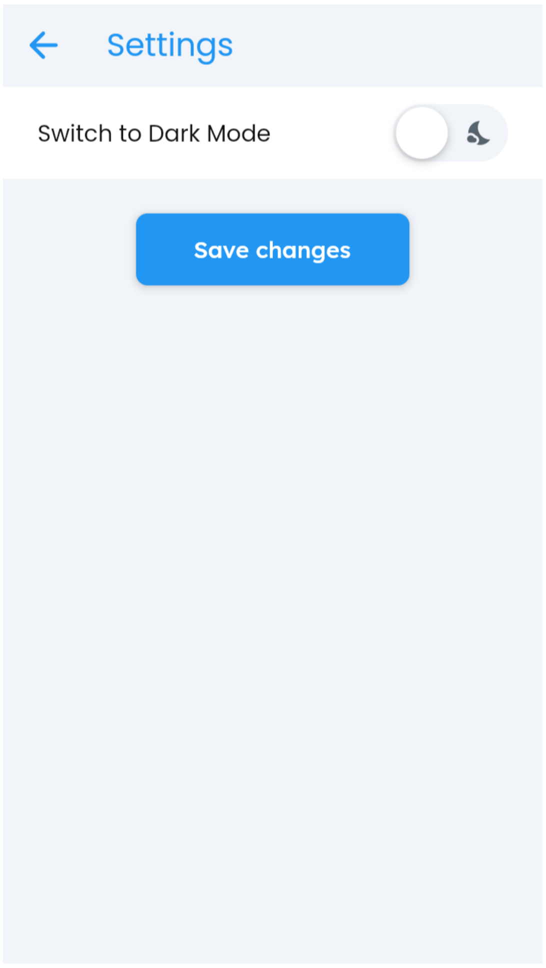


Figure 32: Settings - light mode

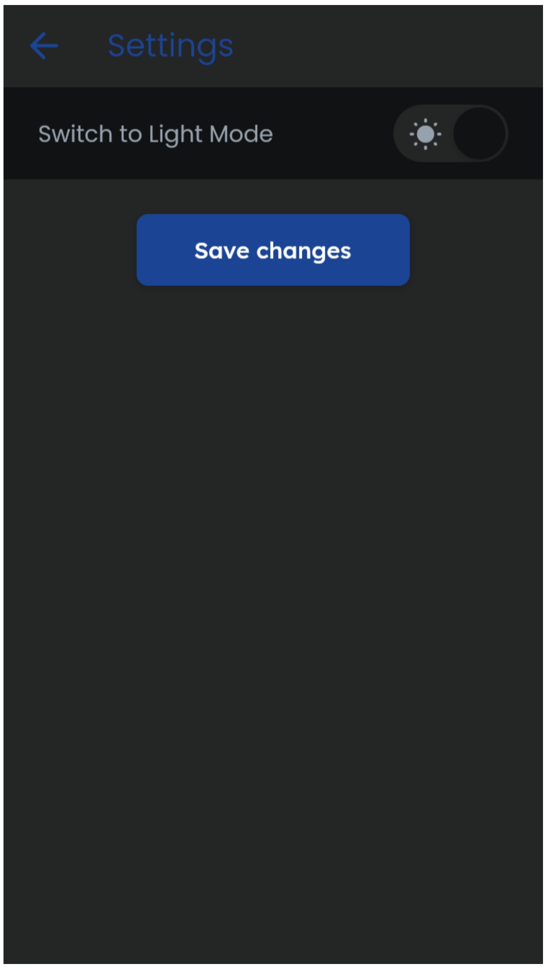


Figure 33: Settings - dark mode

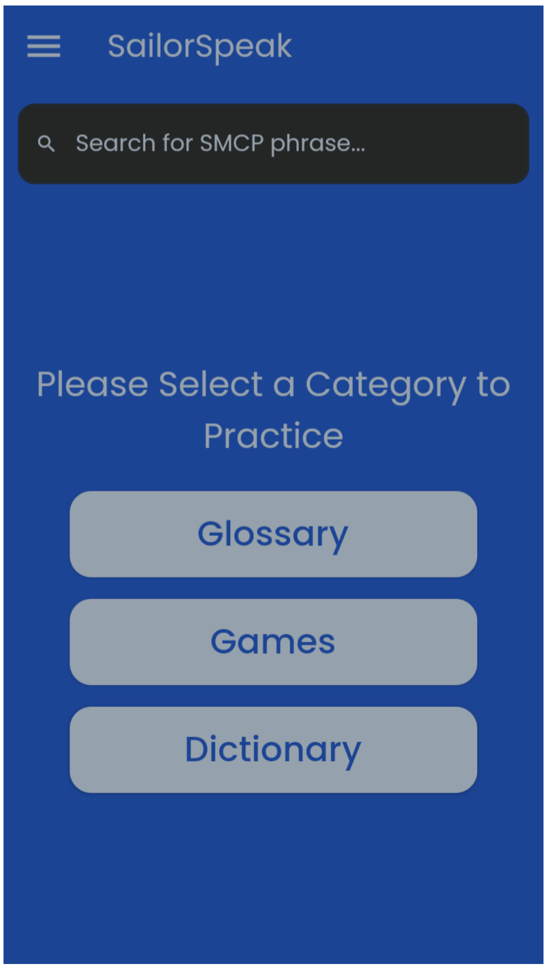

SailorSpeak
Search for SMCP phrase...
Please Select a Category to Practice
Glossary
Games
Dictionary

Figure 34: Homepage Flutter in Dark Mode

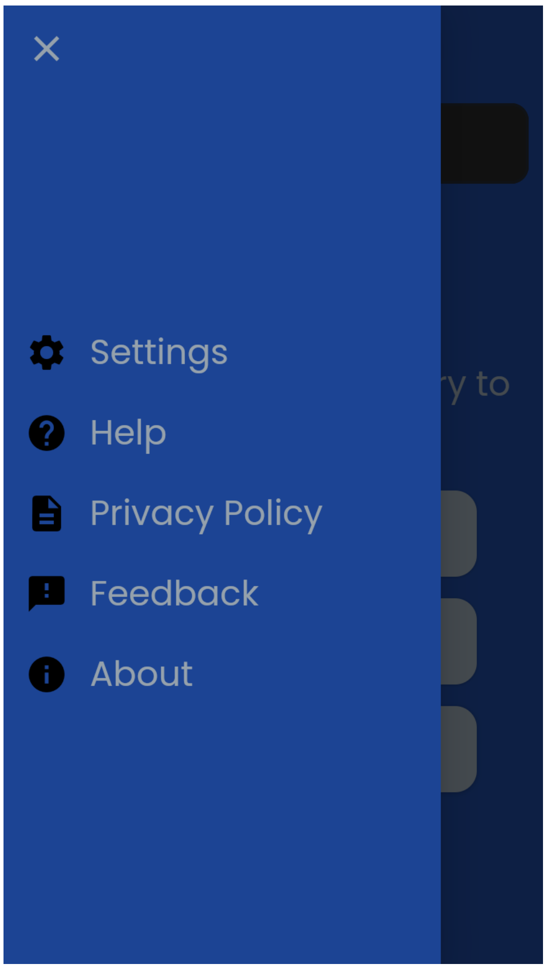


Figure 35: Home menu in Flutter Dark mode

5.4.3 Glossary

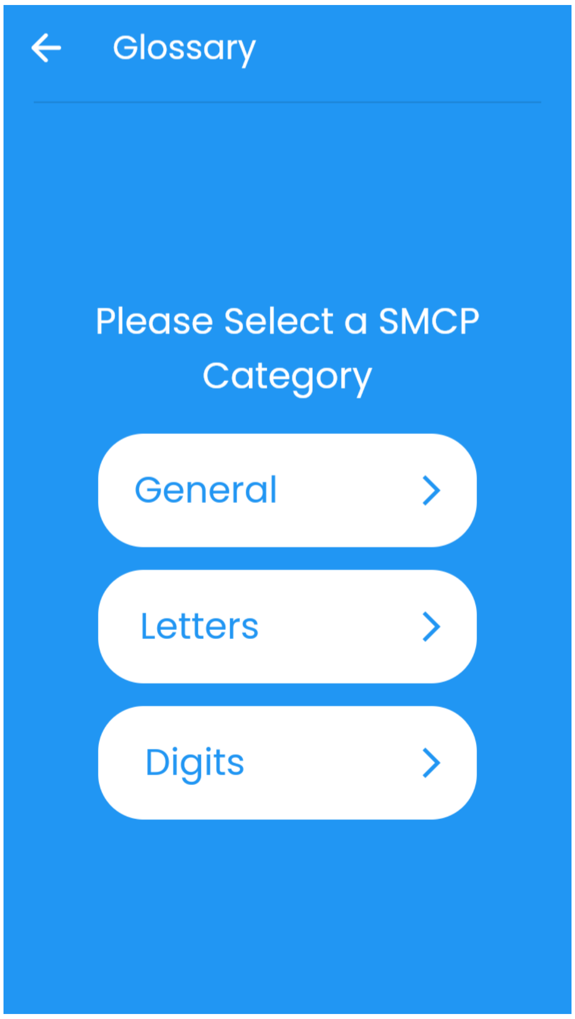


Figure 36: Glossary menu page

The glossary menu was further divided up into 3 subsections:

- **General:** These were general terms taken from the SMCP glossary
- **Letters:** Since letters in SMCP need to be spelled out according to the NATO phonetic alphabet, this needs to be practiced by the speaker.
- **Digits:** Digits in SMCP have a slightly modified pronunciation and therefore needs to be practiced and learned by the speaker.

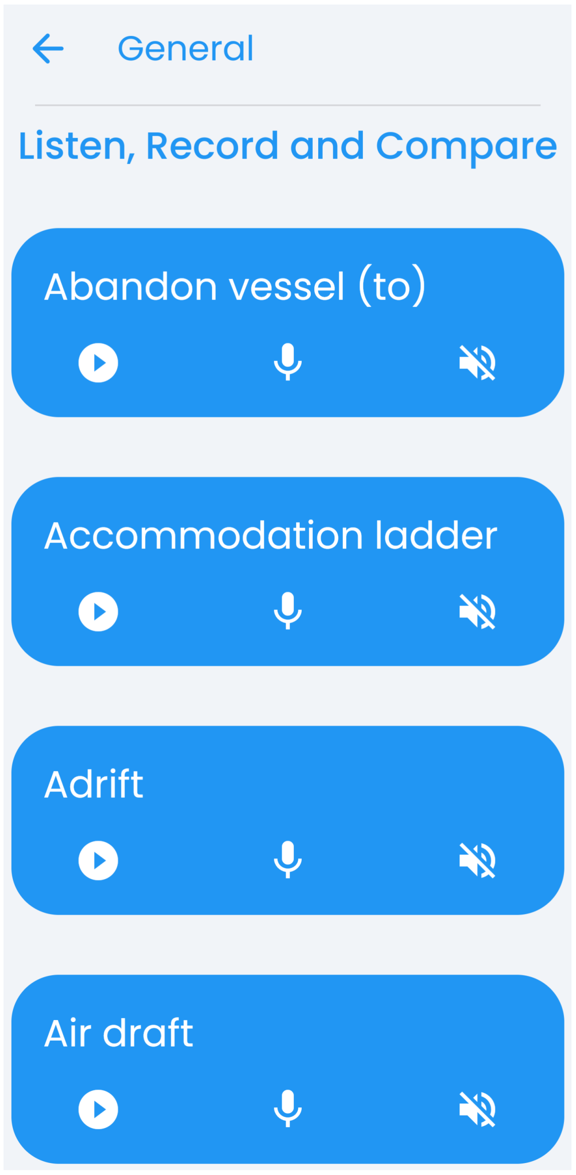


Figure 37: General terms practice page in Glossary

The practice cards had 3 buttons which were:

- **Play button:** Listen to the Native Speaker's pronunciation of the phrase
- **Record button:** Record your own pronunciation of the phrase
- **Playback button:** Play back your own recording of the phrase to compare it with the native speaker's. This playback button was initially disabled since the user had not recorded their voice yet, but became enabled once the user had recorded their voice for a specific phrase.

From figure 38, we can see the microphone turning red, which happens when a user presses down on it, indicating that it is recording. Figure 39 shows the playback button(3rd button) becoming enabled after the user has recorded their voice for the phrase. By clicking on the playback button, they can now play back their own voice recording.

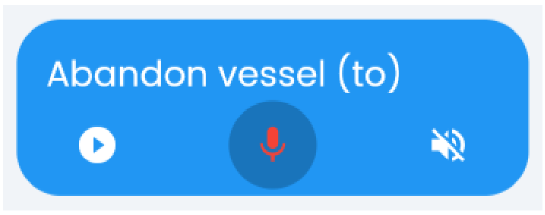


Figure 38: Microphone pressed

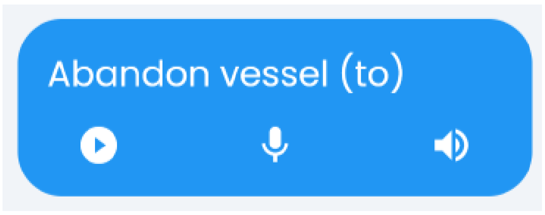
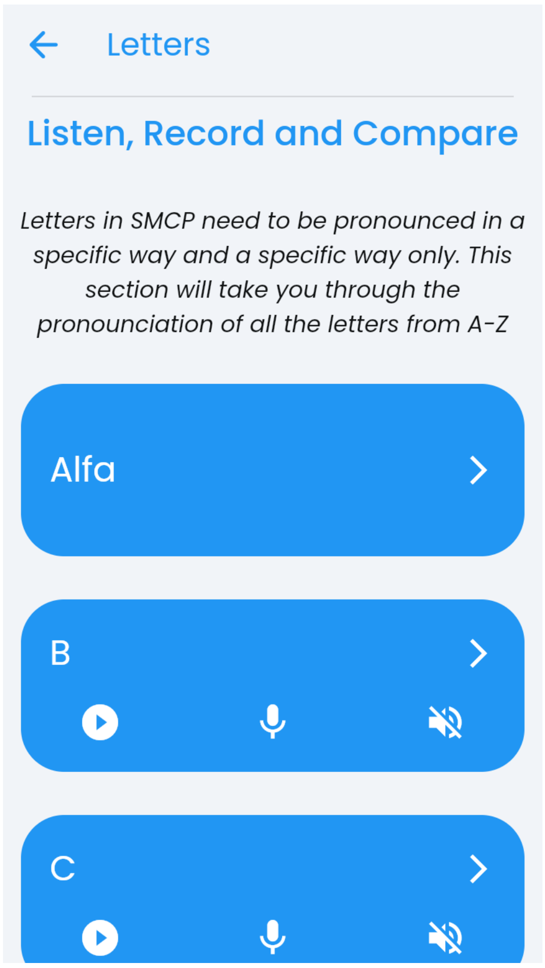


Figure 39: Enabled playback button


Letters

Listen, Record and Compare

Letters in SMCP need to be pronounced in a specific way and a specific way only. This section will take you through the pronounciation of all the letters from A-Z

Figure 40: Letters practice page in Glossary

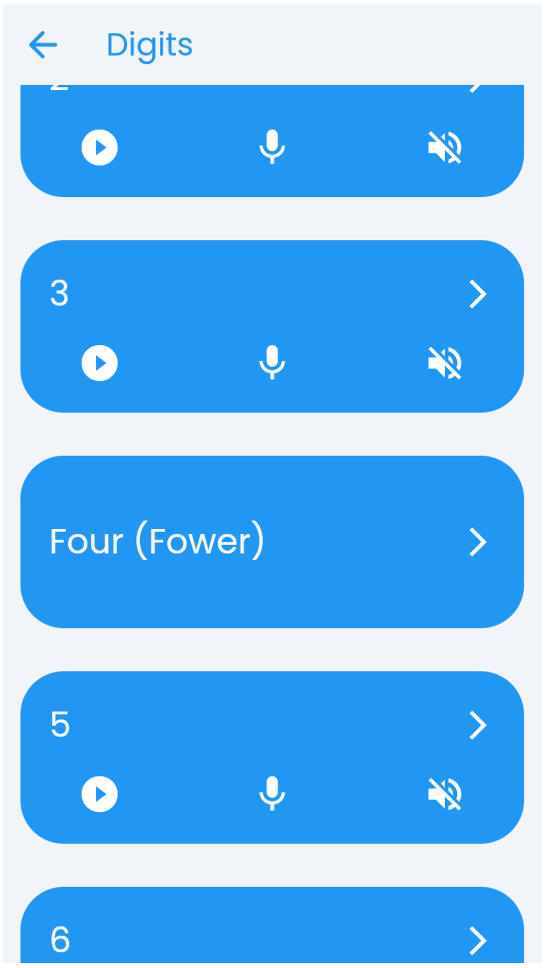


Figure 41: Digits practice page in Glossary

The letters and digits practice page featured a flippable practice card. This allowed the user to see the correct pronunciation of the letter/digit. For example, the correct pronunciation of the letter 'A' in SMCP is "Alpha" (see Figure 40). The correct pronunciation of the digit 4 in SMCP is "fower" (see Figure 41).

5.4.4 Games

Games were another crucial part of this application, which both students and teachers preferred to have(see 5.1.3). Our Flutter application featured 3 games; a quiz, a ship game, and a missing word game.

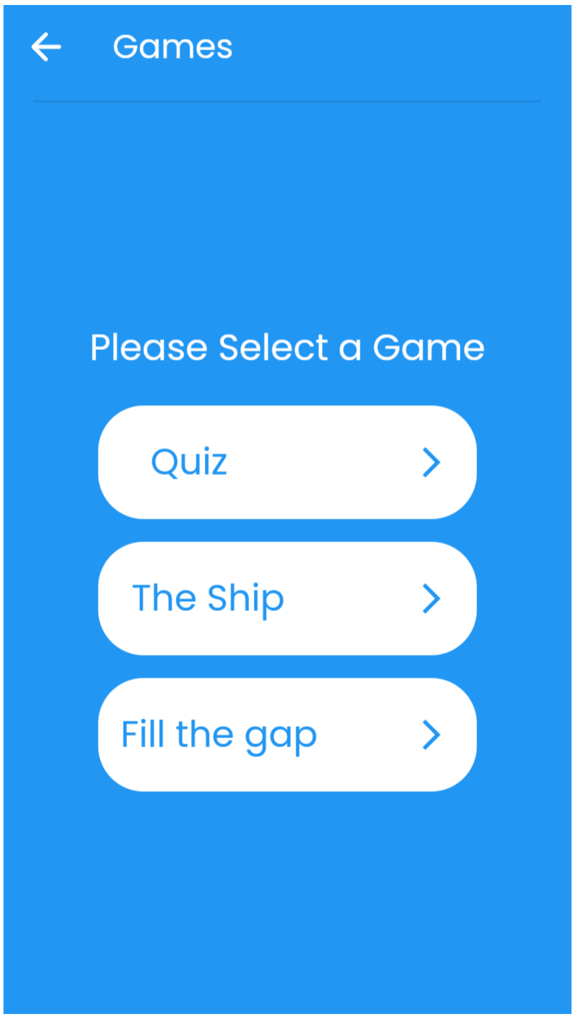
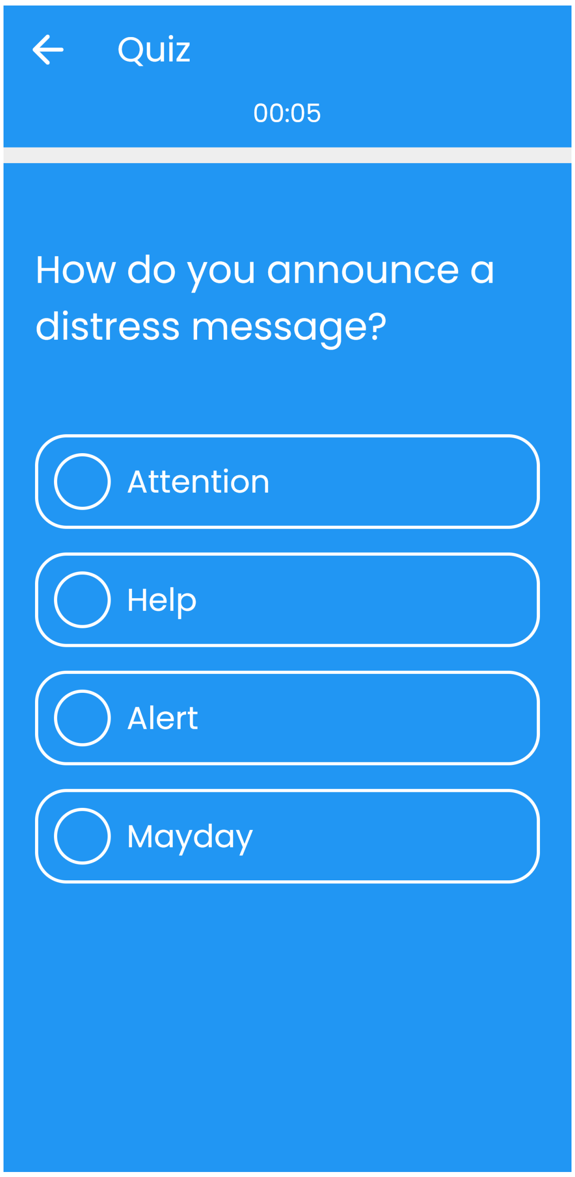


Figure 42: Games menu page

5.4.4.1 Quiz

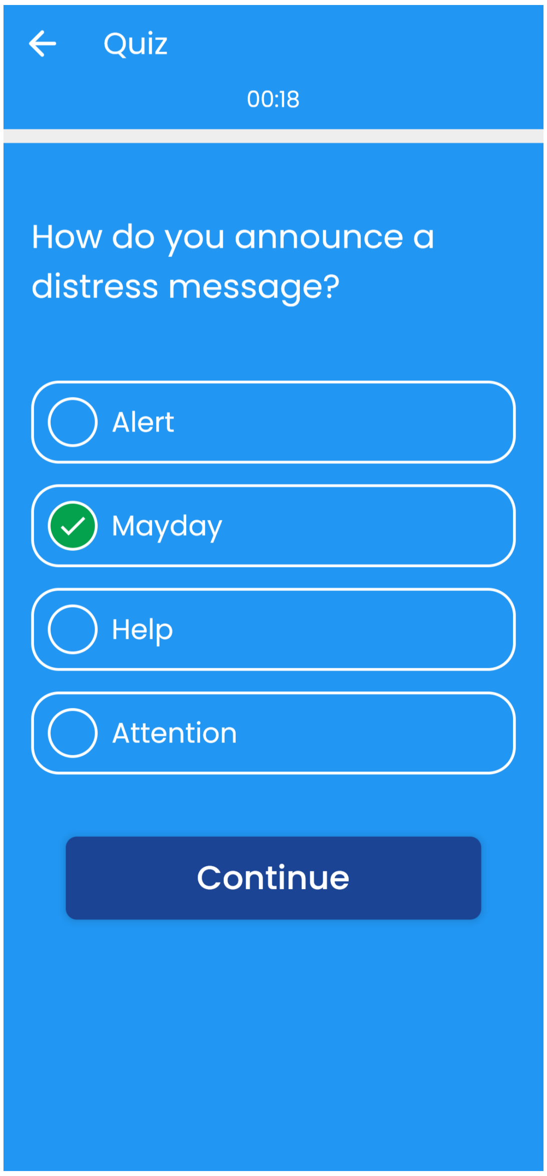
The quiz consisted of 4 questions related to SMCP, with each question having 4 possible options where 1 option was correct. At the end of the quiz, the user got to see their score, as well as the time it took for them to complete the quiz together with an animation of a moving ship.



How do you announce a distress message?

- Attention
- Help
- Alert
- Mayday

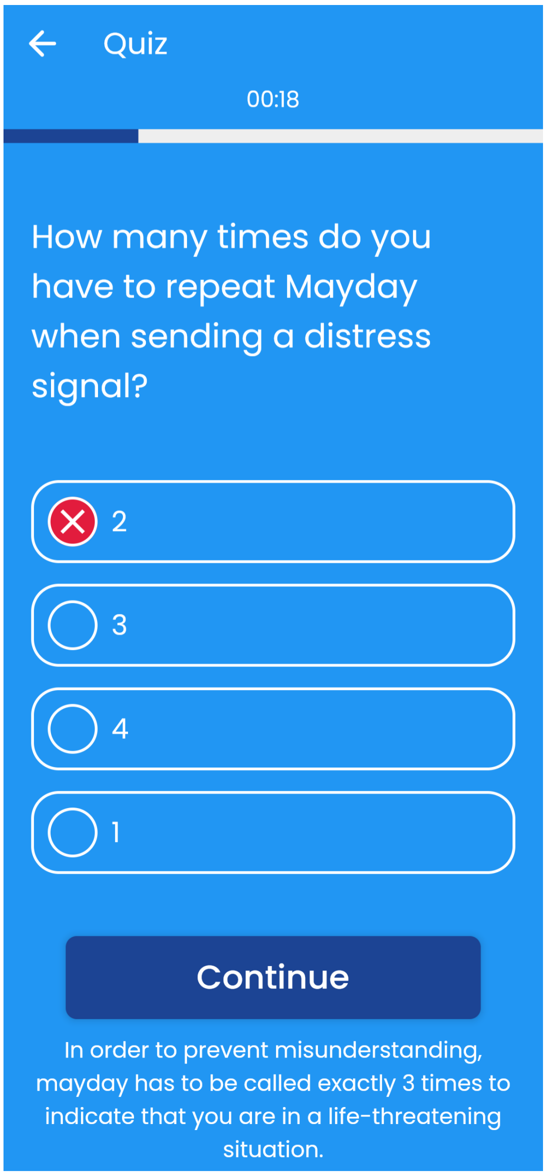
Figure 43: Quiz first question



How do you announce a distress message?

- Alert
- [✓ Mayday]
- Help
- Attention

Figure 44: Quiz correct answer selected



How many times do you have to repeat Mayday when sending a distress signal?

- [✗ 2]
- 3
- 4
- 1

In order to prevent misunderstanding, mayday has to be called exactly 3 times to indicate that you are in a life-threatening situation.

Figure 45: Quiz wrong answer selected

From figure 44, we can see that a green checkbox appears when the user selects the correct answer. The continue button is revealed which lets the user move on to the next question of the quiz. When the user selects a wrong option, as seen in figure 45, there is a red cross as well as an explanatory text below the continue button which informs the user about the correct answer. This is so that the user learns something while taking the quiz, instead of just knowing whether they got the right or wrong answer.

An inconvenience occurred in the quiz for smaller phone screens which was that the user had to scroll down to see the continue button/the explanatory text.

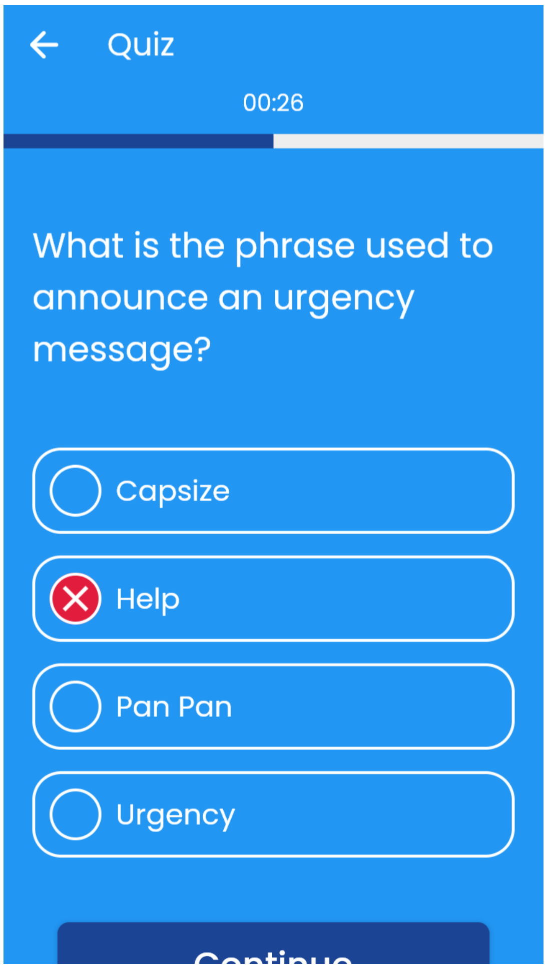
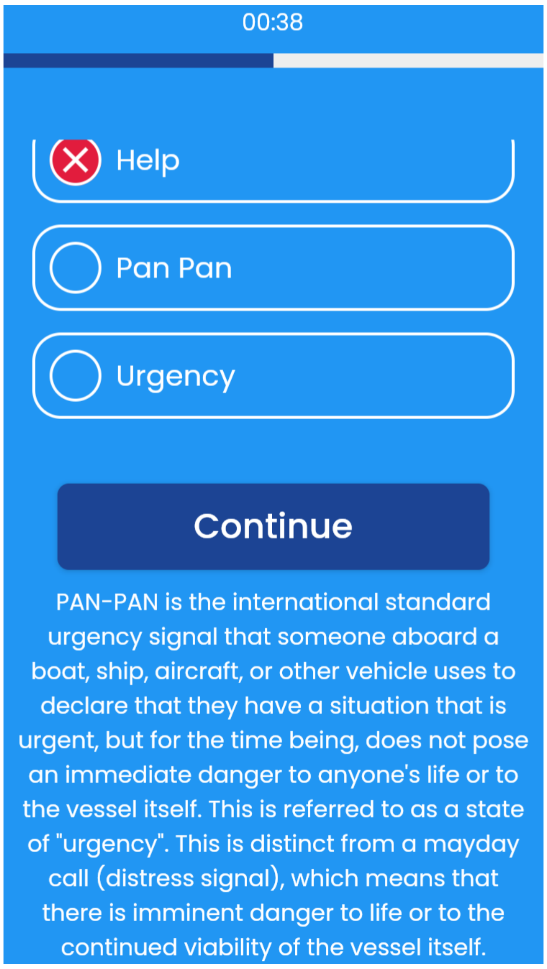


Figure 46: Quiz continue button & text hidden

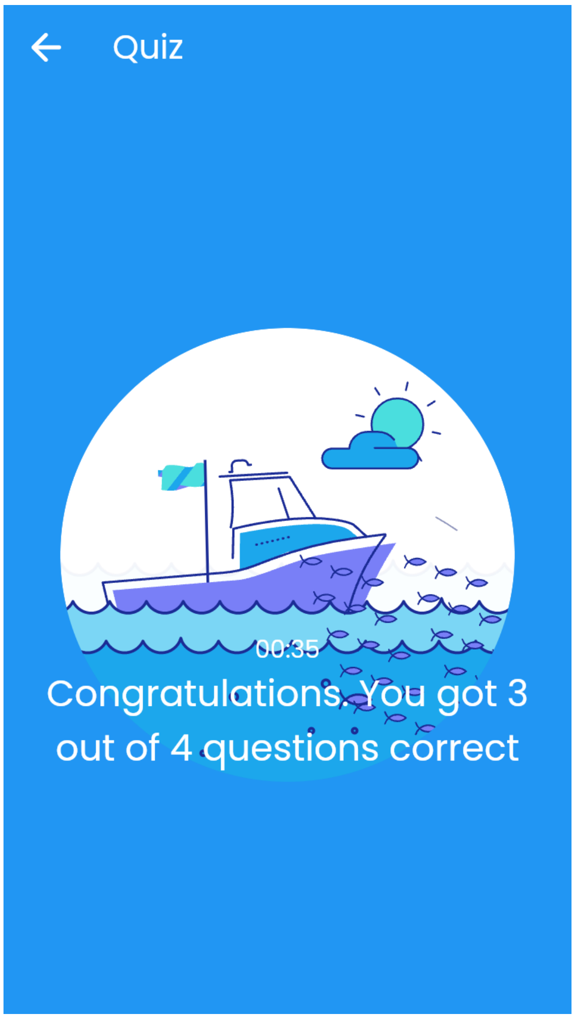


PAN-PAN is the international standard urgency signal that someone aboard a boat, ship, aircraft, or other vehicle uses to declare that they have a situation that is urgent, but for the time being, does not pose an immediate danger to anyone's life or to the vessel itself. This is referred to as a state of "urgency". This is distinct from a mayday call (distress signal), which means that there is imminent danger to life or to the continued viability of the vessel itself.

Figure 47: Quiz question & options hidden

From figure 46, we can see that the continue button is partly hidden after the user has selected an option. However, the explanatory text below the continue button is fully hidden. When the user scrolls down to see the continue button/explanatory text (Figure 47), then the question and options disappear out of view. This created a less friendly user experience in our opinion, since it forced the user to scroll down in order to go to the next question, which should not have been necessary. When creating the quiz in Flutterflow, this issue was not so clear since the phone screen used in Flutterflow for testing is larger. However, when testing on smaller screen sizes, it became evident that this is an issue.

One solution to this issue could be to reduce the font size of the question as well as the size of the quiz answer options.



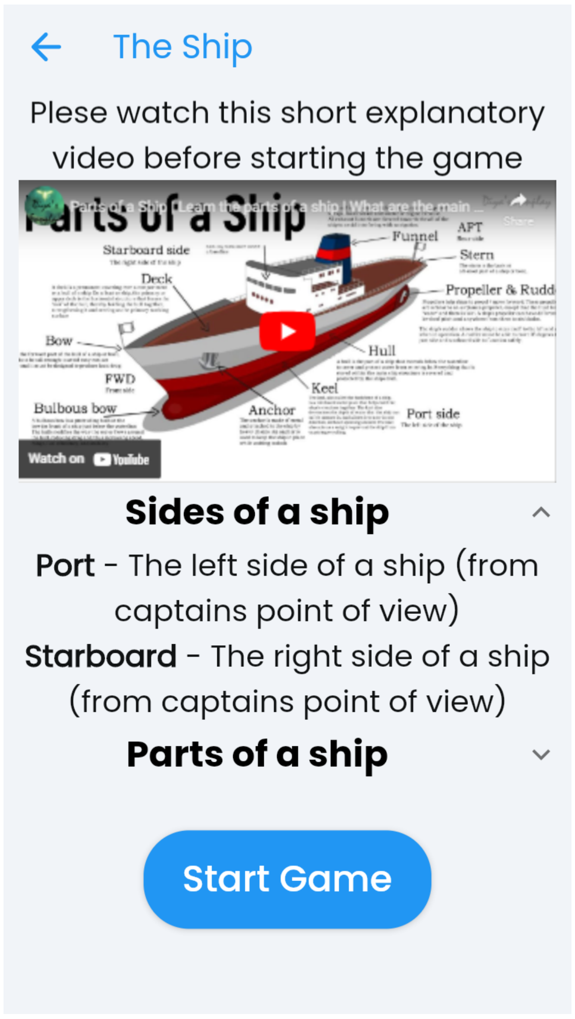
Congratulations, You got 3 out of 4 questions correct

Figure 48: Quiz completion page

Once the user completed the quiz, he was presented with a congratulation page that showed the number of correct answers, the time it took to complete the quiz as well as a colorful animation of a moving ship.

5.4.4.2 The Ship Game

This game consisted of an image of a ship with different parts of the ship labeled with different numbers. Below the ship was a list of the different parts of a ship. To complete the game, the user had to order the different parts of the ship chronologically according to the numbered labels on the ship image. The need for such a game was highlighted in 5.1.3.


The Ship

Plese watch this short explanatory video before starting the game

Sides of a ship

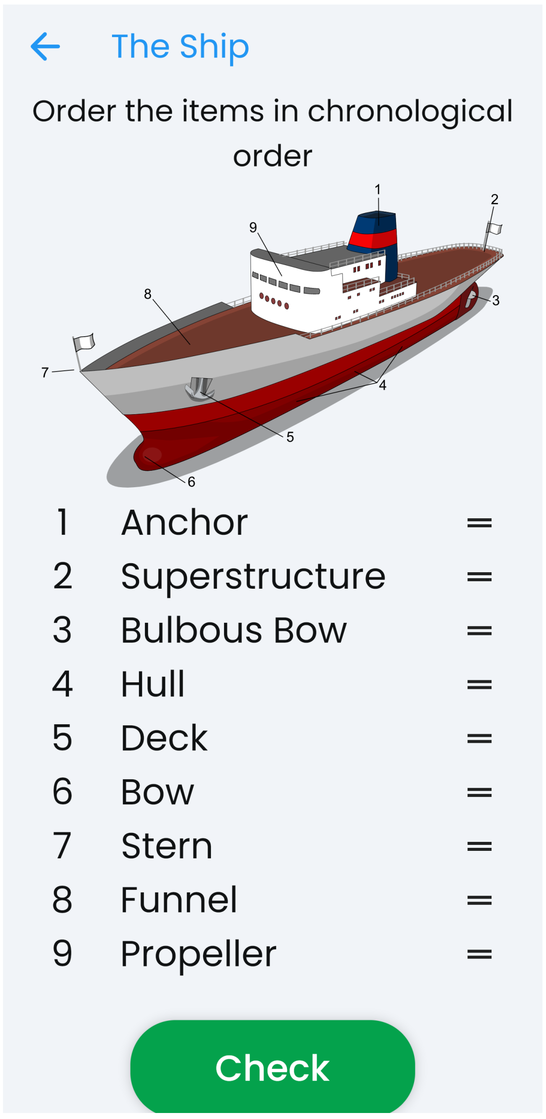
Port - The left side of a ship (from captains point of view)

Starboard - The right side of a ship (from captains point of view)

Parts of a ship

Figure 49: The Ship Game page 1

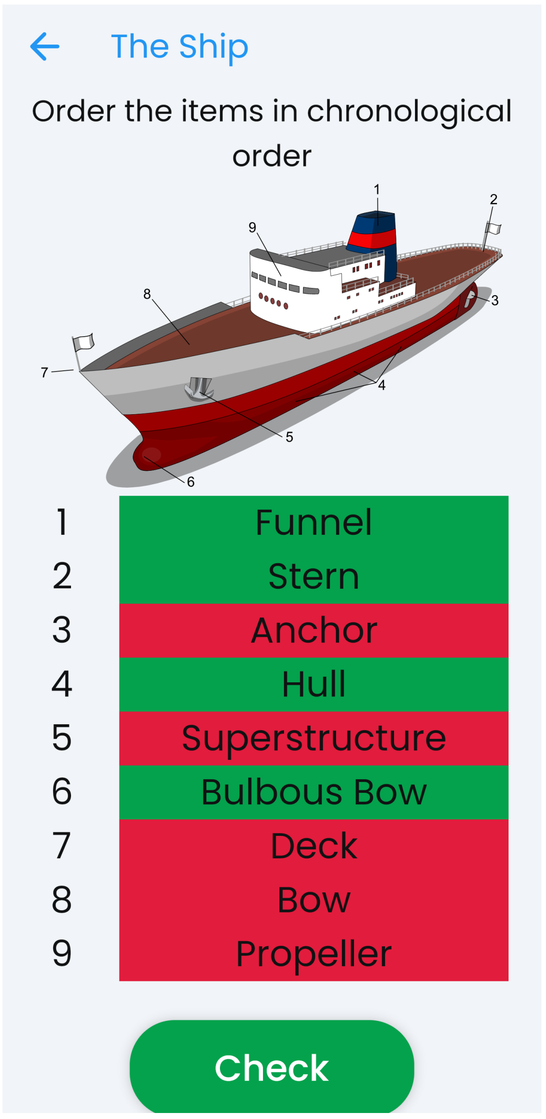
The first page of the ship game was aimed as a learning aid to support the exercise. This was so that the user had the chance to learn the different parts of the ship before they started the actual game. For this purpose, a video was provided as well as explanatory text for the different parts of the ship. The explanatory text was wrapped inside of an Expandable widget, which was initially collapsed. By clicking on the downwards facing arrow, the text inside the Expandable widget would expand and reveal itself. The reason for doing it this way was so that the user would see the start game button without having to scroll down when they first opened the page. This would make the page more user friendly and less cluttered. This was something that was emphasized by the users in the evaluation phase (see 5.3.2).


The Ship

Order the items in chronological order

1	Anchor	=
2	Superstructure	=
3	Bulbous Bow	=
4	Hull	=
5	Deck	=
6	Bow	=
7	Stern	=
8	Funnel	=
9	Propeller	=

Figure 50: The Ship Game page 2 - Game page


The Ship

Order the items in chronological order

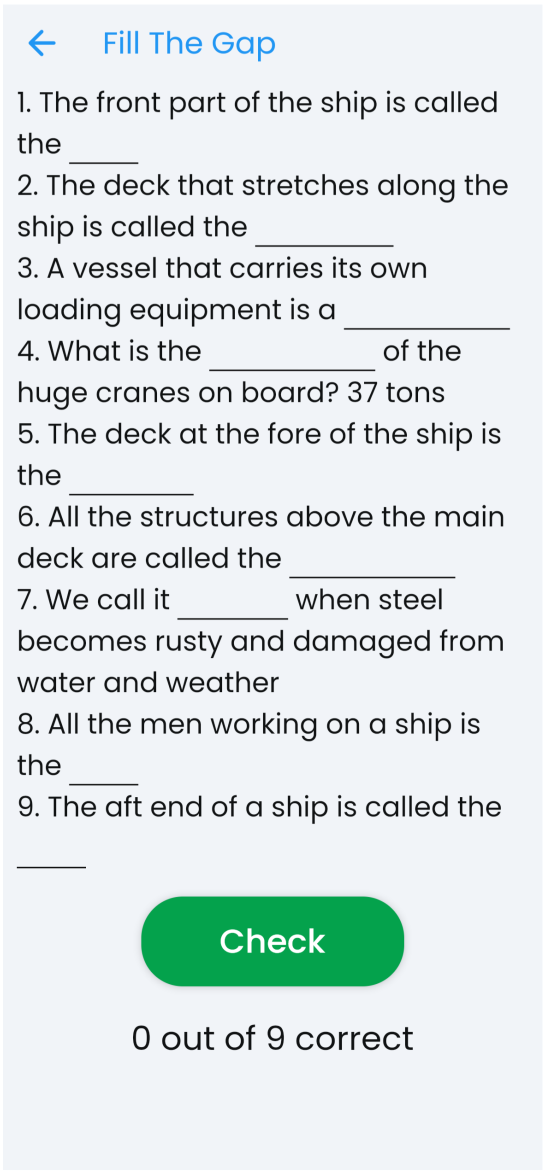
1	Funnel	🟩
2	Stern	🟩
3	Anchor	🟥
4	Hull	🟩
5	Superstructure	🟥
6	Bulbous Bow	🟩
7	Deck	🟥
8	Bow	🟥
9	Propeller	🟥

Figure 51: The Ship Game - Game page checked

Figure 50 depicts the actual game page. The drag handles to the right of the ship parts allows the user to reorder them in the correct order (as marked on the ship). Figure 51 shows what happens when the user presses the "Check" button. The words that are in the correct position are highlighted with a green background and the words that are in the wrong position are highlighted with a red background. For example, the word funnel is placed in position 1 of the list and the funnel of the ship in the image is marked with a 1. Therefore, the word is correctly placed and given a green background when the user presses the check button.

5.4.4.3 Missing Word Game

The Missing Word game consisted of 9 sentences with missing gaps in them. The user had to fill in the missing gaps with the correct word/phrase. This allowed the user to practice SMCP in a contextual way.



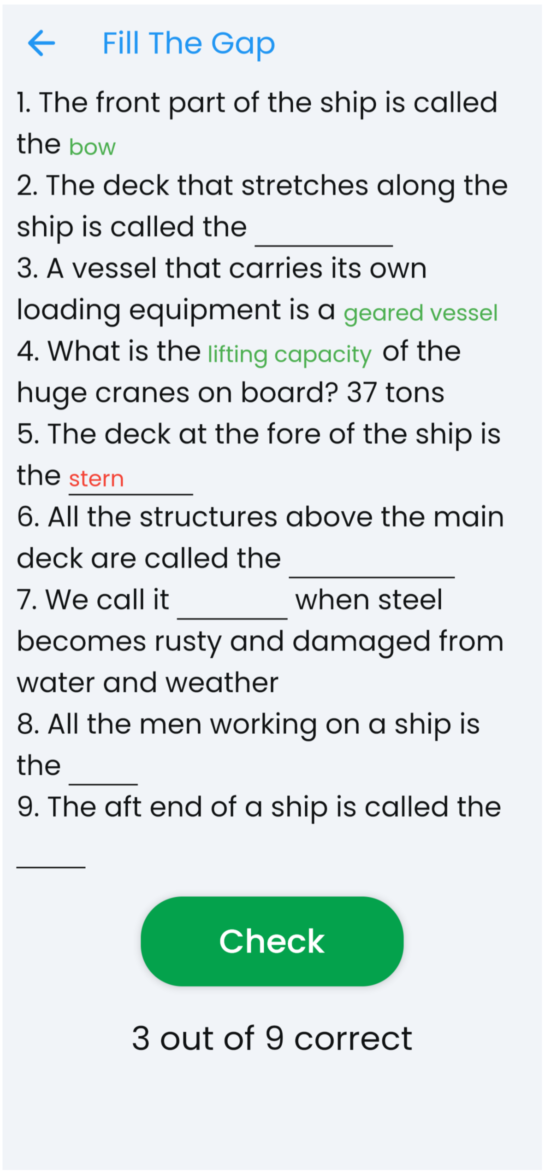
Fill The Gap

1. The front part of the ship is called the ____
2. The deck that stretches along the ship is called the ________
3. A vessel that carries its own loading equipment is a ________
4. What is the ________ of the huge cranes on board? 37 tons
5. The deck at the fore of the ship is the ________
6. All the structures above the main deck are called the ________
7. We call it ______ when steel becomes rusty and damaged from water and weather
8. All the men working on a ship is the ____
9. The aft end of a ship is called the ____

Check

0 out of 9 correct

Figure 52: Missing word game



Fill The Gap

1. The front part of the ship is called the bow
2. The deck that stretches along the ship is called the ________
3. A vessel that carries its own loading equipment is a geared vessel
4. What is the lifting capacity of the huge cranes on board? 37 tons
5. The deck at the fore of the ship is the stern
6. All the structures above the main deck are called the ________
7. We call it ______ when steel becomes rusty and damaged from water and weather
8. All the men working on a ship is the ____
9. The aft end of a ship is called the ____

Check

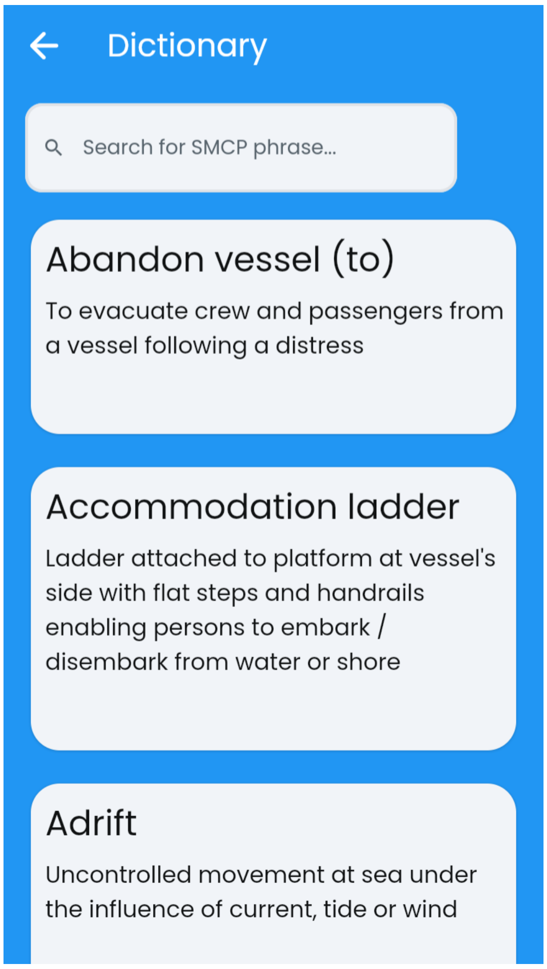
3 out of 9 correct

Figure 53: Missing word game checked

Due to the font size of the text, and also the number of sentences, the full game would not be visible on smaller phone screens. This caused an unfriendly user experience, as the user would have to scroll down to press check, scroll up again to see if their word was correct(green) or incorrect(red) etc. This could be made more responsive by using less sentences per page or changing the font size.

5.4.5 Dictionary

The dictionary featured a searchable list of all of the terms in the SMCP glossary together with their definition.



Abandon vessel (to)

To evacuate crew and passengers from a vessel following a distress

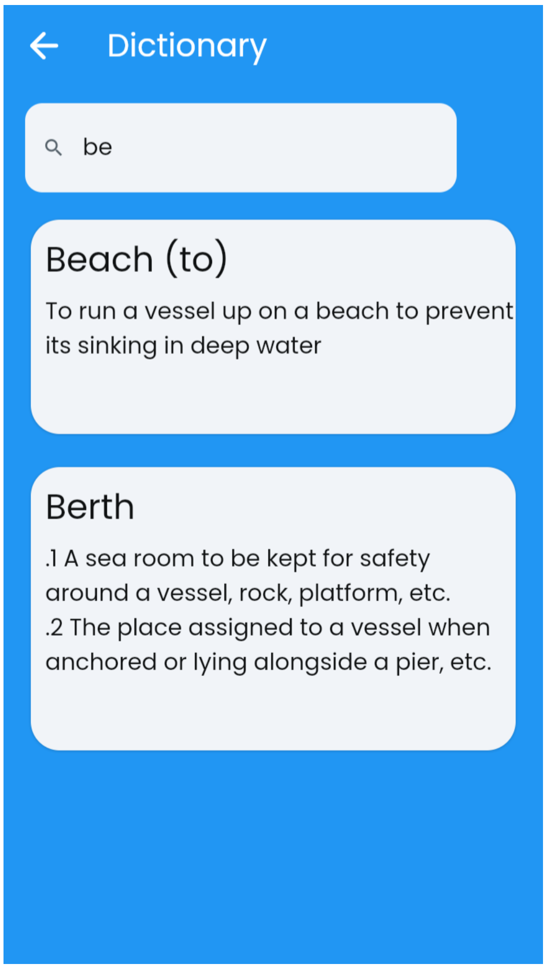
Accommodation ladder

Ladder attached to platform at vessel's side with flat steps and handrails enabling persons to embark / disembark from water or shore

Adrift

Uncontrolled movement at sea under the influence of current, tide or wind

Figure 54: Dictionary



Beach (to)

To run a vessel up on a beach to prevent its sinking in deep water

Berth

.1 A sea room to be kept for safety around a vessel, rock, platform, etc.
.2 The place assigned to a vessel when anchored or lying alongside a pier, etc.

Figure 55: Dictionary filtered with search

The SMCP glossary in the dictionary was ordered in alphabetical order. Figure 55 shows how the results in the dictionary are filtered based on the search term. The search term "be" will update the dictionary to only display phrases that start with "be" which in this case is "Beach" and "Berth".

The search bar in the homepage could further be used to search the dictionary. It worked by autocompleting the search term with phrases in the dictionary that started with that search term. When clicking on the autocomplete option, it would redirect the user to the dictionary with the autocomplete term filled in the dictionary search bar.

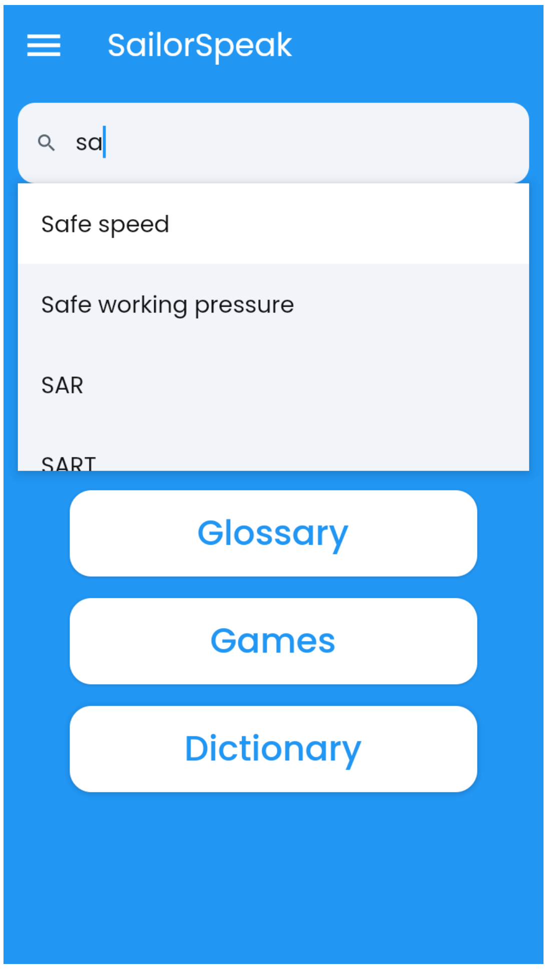
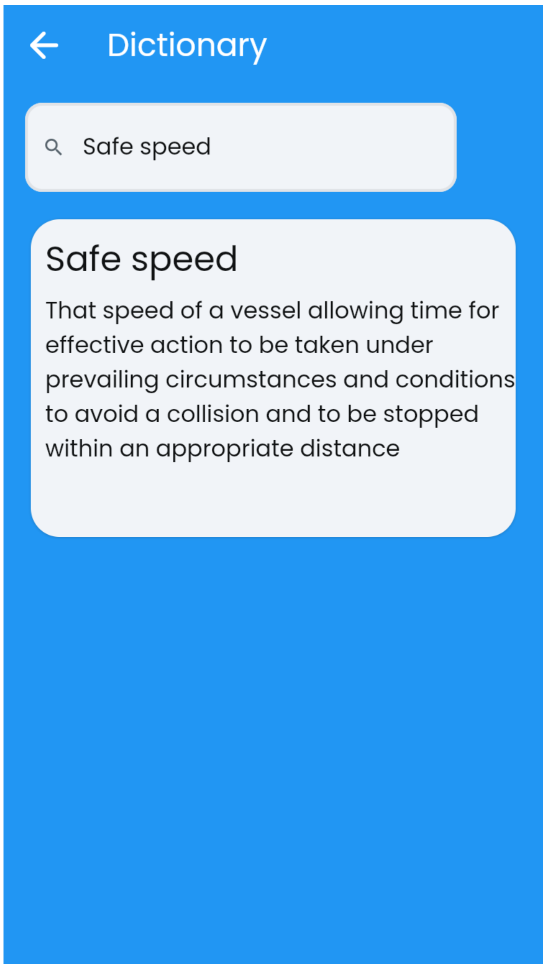


Figure 56: Homepage search autocomplete



Safe speed

That speed of a vessel allowing time for effective action to be taken under prevailing circumstances and conditions to avoid a collision and to be stopped within an appropriate distance

Figure 57: Dictionary redirect

In the example above, we can see the user searching for a phrase in the home page search bar. Figure 56 shows how all of the phrases in the dictionary that start with "sa" show up. The user then clicks on the first autocomplete option which is "Safe speed". Figure 57 shows how the user is redirected to the dictionary page with the term "Safe speed" filled in the dictionary search bar and the results filtered accordingly.

6
Discussion

In this section, we will present the discussion and analysis of the completed project, explore potential areas for further development and consider the ethical and ecological aspects.

6.1 Flutterflow

Using Flutterflow was a great advantage and take-away from this project. It helped speed up the development process and facilitate a more natural translation of our Figma design to our application than would have been possible by manually coding everything. However, there were some limitations on what was possible to do with Flutterflow.

6.1.1 Limitations

Being a low-code builder that is based on Flutter, there were times that Flutterflow lacked functionality or widgets that were available in Flutter. Furthermore, there were situations when we needed custom logic, functionality and code, for example when creating our microphone recorder, that was not available by default in Flutter itself. Hence, these limitations could be classified into 2 categories - Flutterflow limitations and Flutter limitations. Flutterflow limitations were a natural result of Flutterflow being a good, yet not exhaustive extension of Flutter. Flutter limitations were due to the nature of Flutter itself. Many functionalities in Flutter, such as recording a users voice, come from third-party libraries, called packages and are not built into Flutter itself.

2 approaches were employed to address these limitations. The first was to use Flutterflow's manual code editor - which allowed for custom code in the form of custom functions, custom widgets and custom actions. The second approach was to clone the GitHub repository of the project, and work on the manual code adjustments in Visual Studio Code. Both approaches had their advantages and disadvantages. By coding in Flutterflow's manual code editor, it allowed us to directly integrate the custom code into our Flutterflow project and see the changes in Flutterflow. However, this approach had its own limitations. First of all, even though custom widgets were coded manually by yourself, Flutterflow had some limitations on what parameters the custom widgets could take. For example, a custom widget could not have a function as a parameter. It could have an action as a parameter, but the

action could not have any arguments. This made it impossible for us to implement some of our widgets in the Flutterflow manual code editor, and thereby forced us to use Visual Studio Code. In addition to this, Flutterflow manual code editor only allowed you to create new custom widgets. It did not allow you to modify/add functionality to existing widgets created by the visual builder. This feature makes sense, since it could potentially cause the entire application to crash if people get to freely add error-prone code inside of the working code that Flutterflow's visual builder has generated. However, this prevented us from doing simple changes to the components, such as when we wanted to add a dynamic audio path to our audio player, something that was possible in Flutter but not in Flutterflow. Another example was when we wanted to access the onSelected() callback function of the search bar (TextField widget). However, being limited, Flutterflow only provided access to the onChanged() and onFieldSubmitted() callback function of the search bar. This prevented us from implementing the desired functionality for the search bar in Flutterflow, so we had to do it in Visual Studio Code. Even simple things such as adding a gradient color to the homepage (as seen in the Figma design) was not possible in Flutterflow, so it had to be done outside of Flutterflow. A disadvantage of using Visual Studio Code was that even though we pushed the changes in the code to the official GitHub repository of the Flutterflow project, the changes were not reflected in the Flutterflow project. Flutterflow was connected to the GitHub repository, meaning you could push your code from Flutterflow to the GitHub repository, but the GitHub repository was not connected to Flutterflow. So changes made in the GitHub repository did not affect the Flutterflow project, but remained solely on GitHub.

In conclusion, limitations that could be solved using Flutterflow's manual code editor were solved using the manual code editor. Other Flutterflow/Flutter limitations were solved by coding in Visual Studio Code and pushing the code to the Flutterflow GitHub repository.

6.1.2 Bugs

Coding a large application of course presents the challenges of bugs along the way. Some of the bugs we faced were:

1. **The Ship Game:** One of the bugs which occurred was in the ship game. Once a user had completed the game(sorted all items in the correct order), the game would freeze. By investigating the code, we realized that this bug occurred because we were rebuilding the parent widget inside of the build function of the child widget. This caused an infinite loop which caused the application to freeze and ultimately crash. We solved it by removing the call to setState() which rebuilded the parent widget.

2. **Overflow error:** Another bug which occurred was overflow error. This occurred for the Missing Word game. The content on the page was too long to be displayed in its entirety on the phone screen. Therefore, the text overflowed

at the bottom by a number of pixels on certain phone screens. The solution to the overflow error was to wrap the Missing Word Game widget in a SingleChildScrollView, which allowed the page to be scrollable when the content exceeded the maximum height of the screen.

3. **AudioPlayer:** A third bug which occurred was when we tried to provide the audio player with audio files using the setUrl() method. When trying to play the audio files, no sound was played. The solution was to provide the audio files to the audio player using the .setAsset() method instead. The reason was that the audio files were assets that belonged to the project and therefore the .setAsset() method was the appropriate method to use for the audio player.

4. **Microphone Recorder:** A fourth bug which occurred was that the microphone recorder did not work to record the user voice on an Iphone. We had previously tested on android emulator as well as an android phone, and it had worked fine. It also worked fine on Web. However, when testing on an Iphone, we realized that it does not work for iOS. We still do not know the exact reason for this bug, but we managed to solve it by simplifying our code for recording the user voice. Our code for recording the user voice was unnecessarily complicated. By simplifying and refactoring our code to be as minimal as necessary, the bug was solved and the application could successfully record and playback the users voice on an Iphone.

Thanks partly to the visual builder taking care of much of the coding automatically in the background, we did not have as many bugs as we expected for a project of this scope. However, the bugs that occurred helped us learn more about Flutter and Dart and improve our problem solving and debugging skills.

6.1.3 Flutterflow Support & Community

This project would not have been possible without the great help, documentation and support of Flutterflow and its livid community. The Youtube channel of Flutterflow had many great videos and tutorials that were instrumental to learning the tool. The documentation on the Flutterflow website further helped to solidify the concepts and how to use the widgets. When a bug or issue arised, the community was a great place to ask or get answers to questions other people had asked. For certain questions, we directly addressed the support team of Flutterflow at support@flutterflow.com . They were very helpful and engaged to help solve our problems.

6.2 Ethical & Ecological Aspects

The project, from an ethical and ecological standpoint, does not incur any negative effects. As a learning platform, it does incur some positive effects. However, the following considerations could create more positive effects:

- **Data Privacy and Security:** By taking appropriate measures to protect user data and maintain privacy. Our demo application did not collect any sensitive user information such as name, gender, email etc. However, this could change in the future and therefore it is good to have a strong privacy policy and data protection practices in place.

- **Accessibility:** By striving to make the application accessible to users with diverse abilities. Doing so by incorporating accessibility features, adhering to accessibility guidelines, and conducting usability testing with a focus on accessibility.

- **Sustainable Development Practices:** By employing efficient coding practices and optimized algorithms to reduce the application's resource consumption as well as giving consideration to other factors such as efficiency, minimal data usage, and the application's carbon footprint.

- **Responsible Use of Technology:** By considering the potential impact of the application on users and society. Avoiding the promotion of harmful behaviour or enabling the misuse of the application's features.

By taking these considerations and incorporating them, the application could end up incurring more positive ethical and ecological effects.

6.3 Future Work

While the project achieved its objectives and delivered a functional demo application, the project was made with the intent of future development. There is a range of areas that could be explored for future work and enhancements:

- **Sound Wave Comparison:** A feature that existed in the old application, but which we did not implement into our application is for the user to be able to see a sound wave comparison of their pronunciation of the phrase with the native speakers. This feature could help the user to more clearly see how their pronunciation differs from the native speakers and adjust it accordingly.

- **Points on Pronunciation:** A feature which existed in our design, but which we did not have the time and resources to implement, was for the user to get a numerical score on their pronunciation. Based on this score, they would get a green or red border around the phrase which indicated if the pronunciation was good/acceptable or bad.

- **Content Expansion:** The application was built to accommodate expansion. With the addition of more SMCP phrases beyond the current sample phrases and the implementation of more games is a possibility. This includes more content in the current games, such as more questions in the quiz game, more missing words in the missing word game as well as more parts of the ship.

- **Responsive Application:** Throughout this project, we have learned the importance of creating a responsive application that runs well on different platforms and screen sizes. Still, there is room for more fixes and improvements in this domain.

- **Integration with Backend Services:** Integrating the application with backend services, such as a database or API, could enable dynamic content updates and synchronization across devices.

- **User Feedback and Iterative Improvements:** Gathering further feedback on both the design and usability of the application from users and incorporating their suggestions and insights can lead to more improvements and refinements in the application. By expanding the data collection to target other user groups, such as professionals in the maritime industry, the application can attract a broader user base and accommodate to their specific needs.

- **Deploy to Appstore/Google Play Store:** A final step would be to officially deploy this app to the Appstore/Google Play Store so that people all around the world can use it to practice SMCP. We have already been able to successfully deploy the app to Apple Test Flight and Google Play Testing as well as web(GitHub).

7 Conclusion

7.1 Design

The aim of the design phase of the project was to create an intuitive and visually appealing user interface (UI) that aligned with the project's objectives and user requirements. Through the utilization of various design techniques and tools, the following design results were achieved:

- Identification of the requirements for an application that is better aligned with the current needs of users. This involved gathering user feedback and conducting research to understand user preferences and expectations.

- Development of a comprehensive set of wireframes outlining the structure and layout of the application. These wireframes served as a blueprint for the overall design and functionality of the application, providing a visual representation of its various screens and features.

- Creation of a detailed and interactive UI design using Figma. This involved incorporating visual elements, colours, typography, and intuitive navigation to enhance the overall user experience.

- Evaluation of the design prototype through user feedback gathering. This process helped identify areas for improvement and refinement, ensuring that the final design met the expectations and needs of the intended users.

Overall, the design resulted in an interactive prototype that received strong validation from participants in the evaluation phase, with 100% of participants finding it ready to be developed into an actual app (See Figure 27). The interactive prototype can be found **here**.

7.2 Software Implementation

The software implementation phase focused on translating the design concepts and user requirements into a functional and interactive application. Through the utilization of Flutterflow and manual coding approaches, the following software results were achieved:

1. Working and Functional Application:
    - The software implementation resulted in a working and functional application that successfully supported the sample SMCP phrases.
    - The application was developed to be cross-platform, supporting both web and mobile platforms, including iOS and Android.

2. Implementation of a Design Prototype:
    - The design prototype, created during the design phase, was successfully implemented in the software.
    - The visual elements, layout, and interactive features specified in the design prototype were incorporated into the application, providing users with a consistent and intuitive user interface.

3. Implementation of Specified features:
    - The software implementation included the successful implementation of the features specified in the scope section of the report.
    - The application included several interactive games as well as a dictionary to incorporate user needs.

Overall, the software implementation phase resulted in a working and functional application that fulfilled the specified requirements as well as the user needs, and the results can be found **here**.

7.3 Summary

In conclusion, this project has laid a solid foundation for the development of a more modern and updated SMCP language learning application. Moving forward, there are ample opportunities for future work on the application and enhancements on the ethical and ecological aspects for more positive effects. The application, supported by the findings and insights obtained throughout the project, has the potential to make a significant impact on SMCP learning at Chalmers University of Technology and beyond.

Bibliography

[1] International Maritime Organization, https://www.imo.org/ (accessed Jul. 1, 2023).

[2] "IMO Standard Marine Communication phrases," International Maritime Organization, https://www.imo.org/en/ourwork/safety/pages/standardmarinecommunication phrases.aspx (accessed Jul. 1, 2023).

[3] L. Davis, "What is Agile Project Management? The Ultimate Guide," Forbes, https://www.forbes.com/advisor/business/what-is-agile-project-management/ (accessed Jul. 1, 2023).

[4] G. P. Staff, "Agile vs. Scrum: What's the difference?," Graduate Blog, https://graduate.northeastern.edu/resources/agile-vs-scrum/ (accessed Jul. 1, 2023).

[5] "What is user centered design?," The Interaction Design Foundation, https://www.interaction-design.org/literature/topics/user-centered-design (accessed Jul. 1, 2023).

[6] B. Patel, "What is user interface? key terminology explained," Space Technologies, https://www.spaceotechnologies.com/glossary/tech-terms/what-is-user-interface/ (accessed Jul. 1, 2023).

[7] "What is wireframing? A complete guide," UX Design Institute, https://www.uxdesigninstitute.com/blog/what-is-wireframing/ (accessed Jul. 1, 2023).

[8] K. Bracey, "What is Figma?," Web Design Envato Tuts+, https://webdesign.tutsplus.com/articles/what-is-figma–cms-32272 (accessed Jul. 1, 2023).

[9] "What is cross-platform mobile development?," kotlinlang.org, https://kotlinlang.org/docs/cross-platform-mobile-development.html (accessed Jul. 1, 2023).

[10] “Build apps for any screen,” Flutter, https://flutter.dev/ (accessed Jul. 1, 2023).

[11] “Flutter: What is Dart Programming - Javatpoint,” www.javatpoint.com, https://www.javatpoint.com/flutter-dart-programming (accessed Jul. 1, 2023).

[12] “Dart overview,” Dart, https://dart.dev/overview#platform (accessed Jul. 1, 2023).

[13] “Build beautiful, modern apps incredibly fast!,” FlutterFlow, https://flutterflow.io/ (accessed Jul. 1, 2023).

[14] "Visual studio code - code editing. Redefined", Visual Studio Code, https://code.visualstudio.com/ (accessed Jul. 1, 2023).

[15] “Visual studio code,” docs.flutter.dev, Flutter, https://docs.flutter.dev/tools/vs-code (accessed Jul. 1, 2023).

[16] "What is version control: Atlassian Git Tutorial", Atlassian. [Online]. Available at: https://www.atlassian.com/git/tutorials/what-is-version-control (Accessed: 3rd June 2023).

[17] "What is Git: Atlassian Git Tutorial", Atlassian. [Online]. Available at: https://www.atlassian.com/git/tutorials/what-is-git (Accessed: 3rd June 2023).

[18] "What is github? A beginner’s introduction to github" (2022) Kinsta. [Online]. Available at: https://kinsta.com/knowledgebase/what-is-github/ (Accessed: 3rd June 2023).

A

Appendix: Custom code

This section will provide the custom code that was used during the project. This includes custom code for the games, custom code for the voice recording & playback functionality and custom code for other important functionality in the application. The custom code below does not reflect all of the custom code that was added during the project. Some of the custom code involved changing existing components and this code has not been included here.

A.1 The Ship Game

```
// Automatic FlutterFlow imports
import '/flutter_flow/flutter_flow_theme.dart';
import '/flutter_flow/flutter_flow_util.dart';
import '/custom_code/widgets/index.dart'; // Imports other custom
↪  widgets
import '/custom_code/actions/index.dart'; // Imports custom actions
import '/flutter_flow/custom_functions.dart'; // Imports custom
↪  functions
import 'package:flutter/material.dart';
// Begin custom widget code
// DO NOT REMOVE OR MODIFY THE CODE ABOVE!

class ReorderableViewPage extends StatefulWidget {
  const ReorderableViewPage({
    Key? key,
    this.width,
    this.height,
    this.checked,
    this.allCorrect,
  }) : super(key: key);

  final double? width;
  final double? height;
  final bool? checked;
  final Future<dynamic> Function()? allCorrect;

  @override
  _ReorderableViewPageState createState() =>
   ↪  _ReorderableViewPageState();
}

class _ReorderableViewPageState extends State<ReorderableViewPage> {
  final shipParts = shuffleOptions(FFAppState().shipParts.toList());

  void reorderData(int oldindex, int newindex) {
    setState(() {
      if (newindex > oldindex) {
        newindex -= 1;
      }
      final shipPart = shipParts.removeAt(oldindex);
      shipParts.insert(newindex, shipPart);
    });
  }

  @override
  Widget build(BuildContext context) {
    int nCorrect = 0;
    for (int i = 0; i < shipParts.length; i++) {
      if (shipParts[i] == FFAppState().shipParts.toList()[i]) {
        nCorrect++;
      }
    }
    if (nCorrect == 9) {
      widget.allCorrect!();
    }

    return ReorderableListView(
      primary: false,
      children: <Widget>[
        for (int i = 0; i < shipParts.length; i++)
          Container(
              key: ValueKey(shipParts[i]),
              color: widget.checked ?? false
                  ? ((shipParts[i] ==
                   ↪  FFAppState().shipParts.toList()[i])
                      ? FlutterFlowTheme.of(context).success
                      : FlutterFlowTheme.of(context).error)
                  : Colors.transparent,
              child: Text(shipParts[i],
                  textAlign: TextAlign.center,
                  style:
                   ↪  FlutterFlowTheme.of(context).bodyMedium.override(
                        fontFamily: 'Poppins',
                        fontSize: 24,
                      ))),
      ],
      onReorder: reorderData,
      buildDefaultDragHandles: widget.checked ?? false ? false :
       ↪  true,
    );
  }
}
```

A.2 Voice Recording functionality

```
import 'package:record/record.dart';

/**
 * Class for starting and stopping a recording
 */
class RecordingService {
  final record = Record();

/**
 * Starts the recording. On iphone/android, also checks that the
↪  app has permission to record the users voice.
 */
  Future<void> startRecording() async {
    // Check recording permission
    if (await record.hasPermission()) {
      try {
        //start recording
        await record.start();
      } catch (e) {
        print('Error starting recording: $e');
      }
    } else {
      print('No permission to record audio.');
    }
  }

/**
 * Stops the recording and returns the filepath to the recording
 */
  Future<String> stopRecording() async {
    return await record.stop() ?? ""; // Stop recording and return
     ↪  the path
  }
}
```

A.3 Recording playback functionality

```
import 'package:just_audio/just_audio.dart';

final audioPlayer = AudioPlayer();

Future<void> playRecording(String filePath) async {
  audioPlayer.setUrl(filePath);
  audioPlayer.play();
}
```

A.4 Recording Microphone

```
import 'package:flutter/material.dart';
import
↪  'package:sailor_speak/custom_code/actions/RecordingService.dart';
import
↪  'package:sailor_speak/flutter_flow/flutter_flow_icon_button.dart';
import 'package:sailor_speak/flutter_flow/flutter_flow_theme.dart';

/**
 * This class creates a microphone button that uses the Recording
↪  Service to record a user voice.
 * The microphone button turns red when pressed down, indicating
↪  that it has started recording.
 * The microphone button stops recording when released and updates
↪  the recording path using the setRecordingPath function.
 */
class RecordingMicrophone extends StatefulWidget {
  const RecordingMicrophone({
    Key? key,
    required this.setRecordingPath,
  }) : super(key: key);

  /**
   * Updates the recording path when the microphone button is
↪  released. This path will then be used by the app to access the
↪  recording for playback purposes.
   */
  final Function(String) setRecordingPath;

  @override
  _RecordingMicrophoneState createState() =>
   ↪  _RecordingMicrophoneState();
}

class _RecordingMicrophoneState extends State<RecordingMicrophone> {
  late String path;
  bool _isRecording = false;
  final RecordingService recordingService = RecordingService();

  @override
  Widget build(BuildContext context) {
    //Wrapping the FlutterFlowIconButton in a gesture detector is
     ↪  necessary because the gesture detector can detect whether
     ↪  the button is being pressed down/released.
    return GestureDetector(
      //called when the button is pressed down
      onTapDown: (details) async {
        setState(() {
          _isRecording = true;
        });
        recordingService.startRecording();
      },
      //called when the button is released
      onTapCancel: () async {
        setState(() {
          _isRecording = false;
        });
        path = await recordingService.stopRecording();
        widget.setRecordingPath(path);
      },
      onTapUp: (details) async {
        setState(() {
          _isRecording = false;
        });
        path = await recordingService.stopRecording();
        widget.setRecordingPath(path);
      },
      child: FlutterFlowIconButton(
        borderRadius: 30.0,
        borderWidth: 1.0,
        buttonSize: 60.0,
        onPressed: () {},
        icon: Icon(
          Icons.mic,
          size: 30,
          color: _isRecording
```

```
              ? Colors.red
              : FlutterFlowTheme.of(context).primaryBtnText,
        ),
      ),
    );
  }
}
```

A.5 Search filter

```
bool startsWith(
  String searchTerm,
  String searchFor,
) {
  /// MODIFY CODE ONLY BELOW THIS LINE

  if (searchTerm.isEmpty) {
    return true;
  } else {
    return searchFor.toLowerCase().startsWith(searchTerm.toLowerCase());
  }
  /// MODIFY CODE ONLY ABOVE THIS LINE
}
```

A.6 Quiz answer options shuffling

```
List<String> shuffleOptions(List<String> questionOptions) {
  /// MODIFY CODE ONLY BELOW THIS LINE

  questionOptions.shuffle();
  return questionOptions;

  /// MODIFY CODE ONLY ABOVE THIS LINE
}
```