South East Technological University

Functional Specification

Bridge Puzzle Game App for Elementary OS

Student Name: Ming Kit Choy

Student Number: C00246492

Table of Contents

1

1. Introduction

This document details the functional specifications for the Bridge Puzzle Game development for elementary OS, utilising the Vala programming language. The game is expected to comply with elementary OS's design guidelines and will be available on the Elementary App Store.

To enhance the gaming experience, the game will include cloud integration for storing game progress and high scores, allowing players to continue their progress on any device. Additionally, the game may consist of discretionary and exceptional features, such as time limits or new levels, depending on the project requirements.

Overall, the Bridge Puzzle Game for elementary OS will be a highly engaging and enjoyable gaming experience that will captivate players of all ages.

2. Project Scope

This project aims to create a Bridge Puzzle Game for elementary OS that meets the design guidelines. The game will implement core gameplay, integrate cloud storage for high scores and game progress, and may include additional features such as social features, badges, multiplayer modes, user-generated levels, and automatic level generation.

To achieve this objective, the tasks include developing the game's core mechanics, adhering to design guidelines, and implementing cloud integration. The optional features mentioned earlier may also be added to the game.

The deliverables for this project are a fully functional game, code documentation, user documentation, and submission to the Elementary App Store.

Our desired outcomes for this project are a fully operational game, code and user documentation, and submission to the Elementary App Store for public access.

3. Risks and Assumptions

Some potential risks are associated with the project, such as technical challenges when implementing optional features, potential issues with cloud integration, and the possibility of changes to elementary OS guidelines.

On the other hand, some assumptions have been made while planning the project, such as the availability of Vala development expertise, which is necessary for the project's success, the use of reliable cloud services to ensure the smooth functioning of the project, and adherence to elementary OS guidelines to maintain the consistency and quality of the project,

4. Product Overview

This game is designed to integrate seamlessly with the Elementary OS ecosystem, providing a smooth and uninterrupted gaming experience. The game offers various puzzles for players, each presenting a unique and exciting challenge. Furthermore, the game allows players to store their progress in the cloud, making it easy to continue playing from where they left off, regardless of their device.

5. Context Diagram

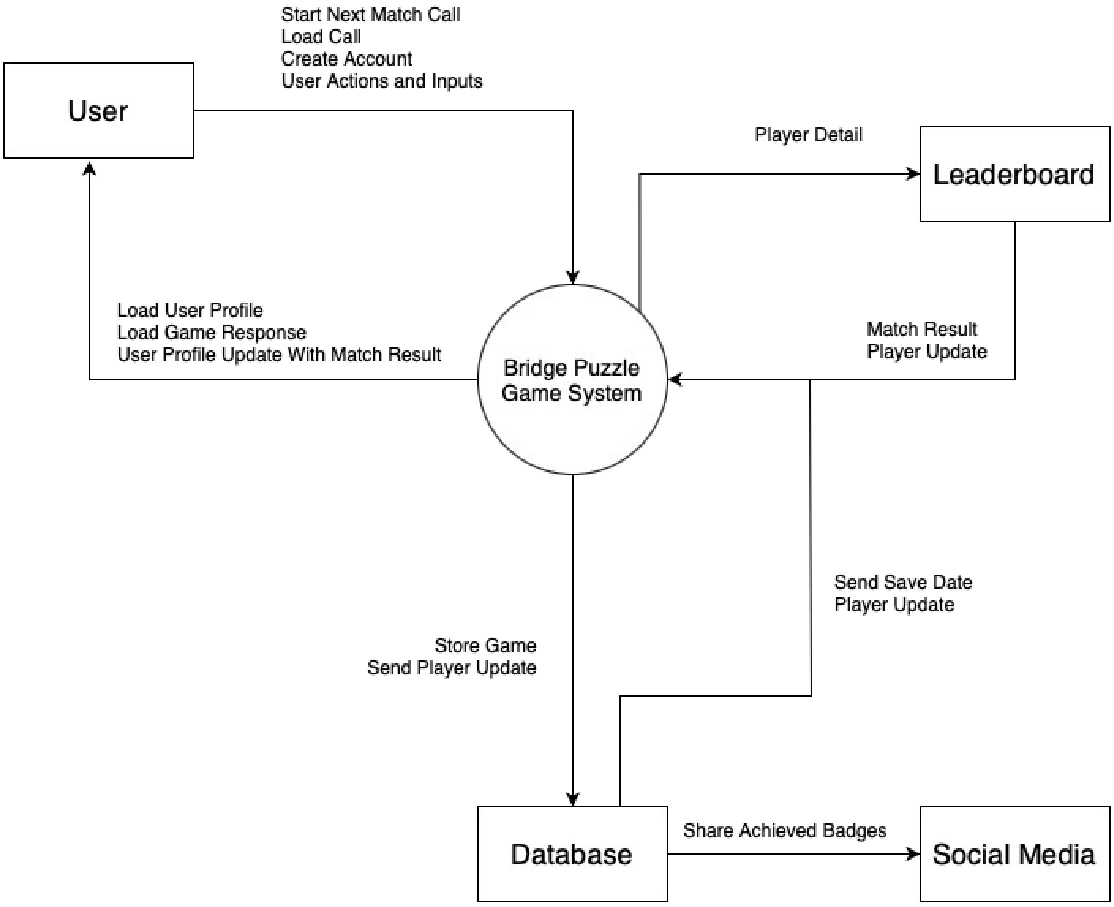

Figure 1 Context Diagram

6.

7. Use Cases

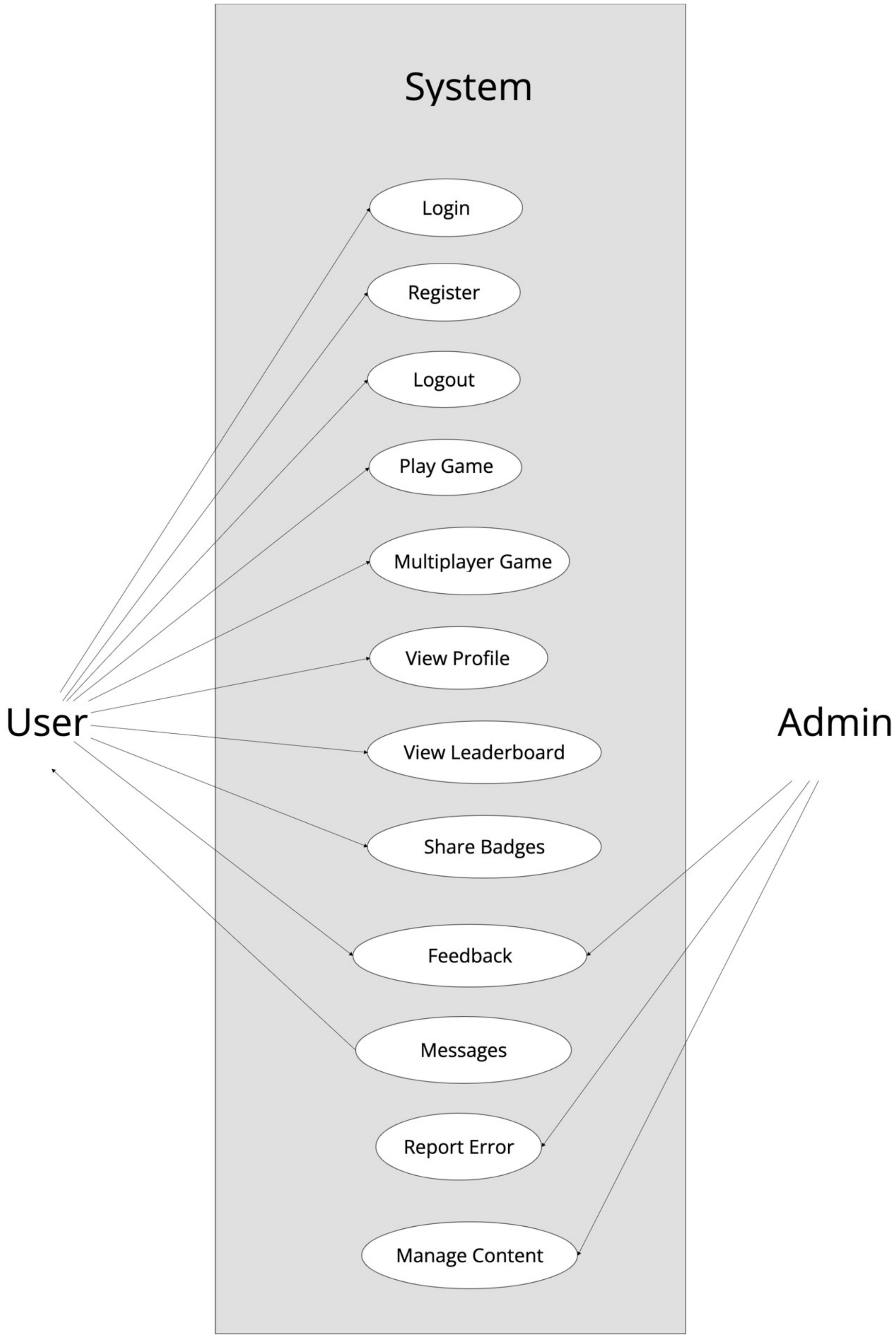

Figure 2 Use Case Diagram

8. Brief Use Cases

Register
Actor: User
Description: This use case begins when a user wishes to enter the game. A user must register an account to gain access to the game. The account creation process mandates the user to furnish a valid email address, devise a unique username, and generate a secure password. The user selects the “register” option. The use case ends when the user has successfully registered into the account.

Login
Actor: User
Description: This use case begins when a user wishes to enter the game. Login allows users to access their accounts by entering their login credentials. The user selects the “login” option. The use case ends when the user has successfully logged in the account.

Logout
Actor: User
Description: This use case begins when a user wishes to log out of the application. The user selects the “logout” option. The use case ends when the vet has successfully logged out.

Multiplayer Game
Actor: User
Description: This use case begins when a user wishes to start the game with another player. The user selects “invite friends”, or “quick match” to start the game. The use case ends when the user and the other player have successfully completed the game.

Play Game
Actor: User
Description: This use case begins when a user wishes to start the game. The user selects different “level modes” of the game. The use case ends when the user has successfully started the game.

View Profile

Actor: User
Description: This use case begins when a user wishes to view their own profile. The user selects the ‘profile’ option to view their achieved badges and leaderboard. The use case ends when the user successfully views the profile.

View Leaderboard
Actor: User
Description: This use case begins when a user wishes to view thier view the leaderboard. The user selects the ‘Leaderboard’ option and view the rank by section of rankings and friends. The use case ends when the user successfully view the leaderboard.

Share Badges
Actor: User
Description: The use case begins when users wish to share their achieved badges from their past game. The user selects the “share” option and shares the badge by social features. The use case ends when the user successfully shares the badge.

Feedback
Actor: User & Admin
Description: The use case begins when a user wishes to provide feedback of the application. The user write a feedback and selects the “submit” option to send. The admin able to access feedback to receive message from users. The use case ends when the user successfully submitted the feedback.

Report Error
Actor: Admin
Description: The use case begins when a admin received a report error from the system. The admin selects the “report error” option to fix. The use case ends when the user successfully fixed the error.

Manage Content
Actor: Admin
Description: The use case begins when a admin wishes to manage content of the application. The admin selects the “manage content” option to edit the content. The use case ends when the user successfully edited the content.

9. Detailed Use Cases

Use Case Name:	**Register**
Actors:	User
Pre-condition:	The application displays the “Register” option.
Main Success Seenarios:	1. The user opens the application. 2. The application requests the register form requesting the users to create a username, and valid email address and create a secure password. 3. The user enters their username, email and password and selects “Register”. 4. Upon successful registration, the user is redirected to the application’s main page.
Post condition:	The user is registered successfully.
Alternative(s):	4a Register unsuccessful • If the register is unsuccessful, the page will display an error message to the user.

Use Case Name:	**Login**
Actors:	User
Pre-condition:	The application displays the “Login” option.
Main Success Seenarios:	1. The user opens the application. 2. The application requests the login form requesting the user’s username and password. 3. The user enters their username, email and password and selects “Login”. 4. Upon successful login, the user is redirected to the application’s main page.
Post condition:	The user is login successfully.
Alternative(s):	4b login unsuccessful • If the login is unsuccessful, the page will display an error message to the user.

Use Case Name:	**Logout**
Actors:	User
Pre-condition:	The user is logged in.
Main Success Seenarios:	1. The user select “Logout” option.
Post condition:	The user is successfully logout.

Use Case Name:	**Play Game**
Actors:	User
Pre-condition:	Login: - The user is logged in. - The “Play Game” option is displayed.
Main Success Seenarios:	1. The user selects the “Play Game” option. 2. The user selects the “Level Mode” option to select different levels of game.
Post condition:	The user successfully starts the game.
Alternative(s):	The user returned to the previous screen.

Use Case Name:	**Multiplayer Game**
Actors:	User
Pre-condition:	Login: • The user is logged in. • The user selected the “Play Game” option. • The “Multiplayer” option is displayed.
Main Success Seenarios:	1. The user selects the “Invite Friends” option or “quick game” option. 2. The user connected with another player by searching for their username or randomly choosing.
Post condition:	The user successfully starts the game.
Alternative(s):	2a No player found • The user return back to pervios page.

Use Case Name:	**View Profile**
Actors:	User
Pre-condition:	Login: - The user is logged in. - The “Profile” option is displayed.
Main Success Seenarios:	1. The user select the “Profile” option. 2. The user able to view their achieved badges and access to leaderboard.
Post condition:	The user successfully enters to their profile.
Alternative(s):	The user returned to the previous screen.

Use Case Name:	**View Leaderboard**
Actors:	User
Pre-condition:	Login: • The user is logged in. • The user selected the "Profile" option. • The "Leaderboard" option is displayed.
Main Success Seenarios:	1. The user select the "Leaderboard" option. 2. The user selects the "Rankings" option to view the highest score in ranking mode. 3. The user selects the 'Friends" option to view the score between the user and their added friends.
Post condition:	The user successfully views the leaderboard.
Alternative(s):	The user returned to the previous screen.

Use Case Name:	**Share Badges**
Actors:	User
Pre-condition:	Login: • The user is logged in. • The user selected the “Profile” option. • The user selected the “Badges” option. • The “Share” option is displayed.
Main Success Seenarios:	1. The user select the “Share” option. 2. The user select the particular “badge” option. 3. The user select the social media features they wish to share to.
Post condition:	The user successfully shares the badges.
Alternative(s):	The user returned to the previous screen.

Use Case Name:	**Feedback**
Actors:	User & Admin
Pre-condition:	Login: • The user logged in. • The user selected the “Setting” option. • The “Feedback & Support” is displayed.
Main Success Seenarios:	User: 1. The user select “Feedback & Support” option. 2. The user select the “Submit” option to submit the message they wish to send. Admin: 1. The admin is able to access the feedback data to read those message send from users.
Post condition:	User: The user successfully submits the feedback. Admin: The admin successfully views the feedback.
Alternative(s):	User: 2a The message area is empty. • The user returned to the previous screen. Admin: 1a The admin details are incorrect. • The admin enters the wrong credentials. • The admin cannot view the “Bridge Puzzle Game” application.

Use Case Name:	**Report Error**
Actors:	Admin
Pre-condition:	The admin views the “Bridge Puzzle Game” application.
Main Success Seenarios:	1. The admin received a report error send from the system.
Post condition:	The admin successfully fix the error.
Alternative(s):	1b The admin details are incorrect. • The admin enters the wrong credentials. • The admin cannot view the “Bridge Puzzle Game” application.

Use Case Name:	Manage Content
Actors:	Admin
Pre-condition:	The admin views the “Bridge Puzzle Game” application. The “Manage Content” is display.
Main Success Seenarios:	1. The admin select the “Manage Content” option.
Post condition:	The admin successfully edited the content.
Alternative(s):	1c The admin selects ‘Cancel’ • The admin does not make any changes. • The admin is returned to the previous screen. 1d The admin details are incorrect. • The admin enters the wrong credentials. • The admin cannot view the “Bridge Puzzle Game” application.

10. Requirements

Core Game Mechanics: Hashiwokeko is a puzzle game that originated in Japan. It involves connecting islands with bridges to form a single connected group of islands. The number on each island indicates how many bridges are required to connect it to the rest of the islands. The game is solved when all islands are related and the number of bridges on each island is correct.

Elementary OS Design Guidelines: the game must adhere to the Elementary OS design guidelines to ensure a consistent user interface and user experience. Following the design guidelines ensures the game looks and feels like it belongs on the Elementary OS platform.

Cloud Integration: the game should have cloud integration, allowing players to store their game progress and high scores remotely. With cloud integration, the players can continue their game progress from any device, making the game more convenient and accessible.

Optional: The game may incorporate social features, badges, multiplayer modes, user-generated levels, and automatic generation.

- Social features like leaderboards and friend lists encourage players to compete with each other and share their progress.
- Badges and achievements reward the players for their accomplishments, adding an extra layer of motivation. Multiplayer modes enable players to collaborate or compete with others, making the game more exciting and dynamic.
- User-generated levels and automatic level generation allow players to create and choose their preferences.

11. Configuration

To set up, users need to gather specific information from each user to set up user accounts for cloud integrations, including their full name, email address, and a secure password. I will then create individual accounts with unique login credentials for each user. These accounts allow users to access the cloud integration platform and utilise its features. Once the accounts are set up, we will provide each user with instructions on logging in and using the platform.

12. Non-Functional Requirements

Non-functional requirements are essential for the success of any software system. In the case of a game, there are specific non-functional requirements that need to be considered.

12.1 Performance optimisation

This is crucial for ensuring smooth gameplay. This includes optimising the game's graphics, animations, and sound effects to ensure they run smoothly on different devices and platforms.

12.2 Secure cloud data storage

As games typically require users to create accounts and store personal information, it is crucial to ensure that this data is stored securely in the cloud. This includes implementing appropriate security measures such as encryption and access controls.

12.3 Responsiveness to user interactions:

This is to ensure an enjoyable gaming experience. This means the game should respond quickly and smoothly to user input, such as button presses or gestures, without noticeable delay. Achieving this requires careful optimisation of the game's code and user interface.

13. Data Security and Privacy

The Bridge Puzzle game prioritizes the security and privacy of user data, implementing various measures for maximum protection. These measures include encrypting all gameplay and user profile data stored in the Firebase NoSQL cloud database with robust encryption methods like AES-256. This safeguards sensitive information such as usernames, passwords, and progress data.

Local device storage uses the SQLite database and employs encryption utilities provided by elementary OS to encode data securely. For password storage, the SHA-256 cryptographic standard and salting are utilized to ensure that passwords selected during user registration and login are stored safely.

To prevent unauthorised access to gameplay information, user access control and Firebase security rules only allow read/write access to users for their data objects. The latest OAuth 2.0 standards are also applied for user authorisation flows for Firebase, with refresh tokens encrypted locally.

All data transfer between the game client and Firebase cloud occurs over HTTPS protocol, using the latest TLS 1.3 specifications for secure channels. In compliance with GDPR policies, players can review and revoke consent to share personal data, with clear terms being displayed during onboarding.

Lastly, the Vala codebase, NoSQL database rules, and encryption practices undergo periodic third-party audits to fix vulnerabilities through the latest patches and updates.

14. Error Reporting

One of the essential features of any application is its ability to handle errors and exceptions gracefully. To achieve this, the application will be designed to display user-friendly error messages that help users understand the issue and how to resolve it. Additionally, the application will provide options for error reporting and resolution, allowing users to report the error and receive assistance from the support team if necessary. This approach ensures that users can quickly and efficiently resolve any issues they encounter while using the application.

15. Release and Deployment Plan

The game app will be developed incrementally, with several release milestones before going public.

15.1 Alpha Testing Phase

An initial prototype with core gameplay logic and 2-3 levels will be completed and made available for internal testing. The focus will be on ensuring the stability of Vala code components and unit testing coverage, as well as validating compliance with Elementary OS UI guidelines and checking cloud integration.

15.2 Closed Beta Testing Phase

A formal beta build with complete gameplay features, 10+ levels, and unlocked achievements will be released on AppCenter for installation by up to 100 test users. Bug reports will be closely monitored, and feedback on UI/UX flow and level difficulty tuning will be gathered via beta user surveys.

15.3 Open Beta Phase

3 languages will be added for support, and the beta will be opened up for anyone to test and submit feedback via AppCenter and community forums. High-priority issues will be addressed, all text will be translated, and graphics and effects will be refined.

15.4 General Availability

Once all of these phases are complete, the game app will be made available to the public on the AppCenter store for all Elementary OS users. An ongoing improvement plan will be implemented, with monthly patches and additional levels released quarterly. The first feature release is tentatively scheduled for 6 months post-launch, which will include multiplayer mode integration among its goals.

16. Conclusion

In conclusion, the functional specification presented in this document provides a detailed plan for developing the Bridge Puzzle Game for elementary OS. The goal is to create an engaging, user-friendly game that integrates seamlessly with the primal OS ecosystem. This will be achieved by adhering to the design guidelines of elementary OS and leveraging the Vala programming language.

Players can continue their progress across different devices by incorporating cloud integration, adding a valuable dimension to the gaming experience. Additionally, the possibility of including optional features such as social elements and multiplayer modes will enhance the game's appeal and replay value.

Although certain risks and assumptions are associated with any development project, careful planning, execution, and adherence to elementary OS guidelines will contribute to the project's success. Ultimately, the Bridge Puzzle Game for elementary OS aims to captivate players of all age groups and offer a fun and challenging gaming experience that aligns with the elementary OS ethos.