2DV609

Requirements Specification

WinnerDrinks

Version 2

2021-05-30

Delfi Šehidić
Joel Martelleur
Lucas Sjölander
Susanna Persson
Caesar Lennartsson

Contents

1. Introduction

1.1 Purpose

The main purpose of this document is to define and document the application *WinnerDrinks'* system requirements, use cases, and domain rules. In addition to this, this document also defines and documents the requirement of the application's user interface and interfaces to external systems and also the client's constraints in the application and in the project.

This document is mostly intended for software engineers who work with this project and with developing the application but can also be read by other stakeholders of the application.

If you are not a software developer in the project and want to get a quick overview of the project and the application, section *1.2 Scope* is recommended to read. Or if you want to know more about different user scenarios in the application, such as choosing a game, starting a game or playing a game, then section *6 Use-Cases* is a good section to read. To get a sense of what the end product itself will look like, section *3.1 User Interfaces* is worth reading. Also, sections *2.4 Module Party Requirements, 2.5 Module Trivia Requirements, 2.6 Module Spin the Wheel Requirements*, *2.7 Module Back to Back* may be of interest to read as these sections address the requirements that the actual game modules themselves shall contain. Of course, everyone is welcome to read all parts of this document to get a deeper insight into this project and what requirements the application shall contain.

1.2 Scope

The scope of this project and application *WinnerDrinks* is to create a fun party game that can be played on a mobile device, reading tablet or desktop. The target groups for the application, the game in the application, the user interface and the feeling of the application as a whole is aimed for small gatherings and parties with around 2-10 individuals, with young adults around 18-30 years. A scenario where the application is used could be in a party where a person starts the application on a phone and enters who wants to play the game and then the game is on. The difference between this application and other similar applications like Piccolo, Ryggio or PartyPal is that the game in this application does NOT hand out punishments to the game participants who lose. Instead the purpose and construction of the game is to get to know each other, have fun together and that it is the winner(s) who get the most drinks, which better reflects the usual drinking setting and takes a fresh approach to drinking games. The game in the application should be easy to understand and you should be able to play it without actually having to read any comprehensive game instructions. The game will be constructed with multiple game rounds, each game round will contain a game event, the participants of the game event and the prize to the winner(s) of the game event. These game events can be questions, challenges, competitions or random winners that the game participants must succeed, pass, win or get in order to earn their drinks.

1.3 Software development life cycle (SDLC)

The game in the application will consist of separated game modules. These game modules shall be able to be developed in a modular fashion, this means that it shall be possible to add new game modules, remove games modules or update existing games modules without affecting other game modules that already exist in the game. In this way, it will be possible to quickly update the application during the SDLC.

1.4 Agile approach

The application will be developed in an agile approach. This means that early design, implementation and testing of the application will be used as feedback to this document in order to improve the completeness and correctness of the requirements. The absolute core of the requirements will in all probability remain but more requirements and minor changes in some core requirements and use cases are to be expected during the development phase of the application.

1.5 Definitions and abbreviations

Application: The application WinnerDrinks.

Game participant: A participant (player) in *the game*. A game participant can have different states.

1. Active (default): A participant that is included in *the game* but not necessarily included in a specific game event.
2. Inactive: A paused participant that is currently NOT included in the game but could be set to active again.
3. Participating: A participant included in a specific *game event.*

Game Round: One round of a game that typically results in one or more winners. A game round consist of 3 phases:

1. A *game event* phase
2. A winning and prize phase
3. A transition phase to the next round

Game event: A question, challenge, task, competition or a random selector that the game participants should try to succeed, do, win or get. A game event has the responsibility to display the event and to decide the number of participating game participants.

Game event types: The game events in the game are of types *Challenge, Competition, Luck, Question.*

1. Challenge: This game event type offers a challenge for one participant to complete in order to win.
2. Competition: This game event type offers some competition between participants, who compete for the win.
3. Luck: This game event type randomly selects one of the participants as the winner.
4. Question: This game event type presents a question to the participant and the participant chooses between a number of possible answers, where the correct answer is a win.

Game mode: Different versions of the game

1. Standard: Playing with alcohol as a prize.
2. Scoreboard: Playing with a scoreboard without alcohol as a prize.

Game module: A game component containing a set of similar game events. A game module and the game events of that game module are plugged into the game automatically but can also be excluded from the game by the user of the application.

The Game: The actual game *the user* interacts within the application. The game consists of multiple *game rounds* and has the responsibility to keep the total score of the *game participants* and to suggest a price for the winning *participating game participants* in each *game round*.

User: The one person that interacts with the application.

2 System-wide Requirements

2.1 Stakeholders

Primary Stakeholders:

1. User of the application
2. Game participants
    a. People at small parties and
    b. Young adults, age 18-30
3. Software engineers
    a. Software Developers
    b. Software Testers
    c. Software Designers
4. System engineers
    a. Release Engineers
    b. System Administrators
    c. Database Administrators
5. Client

Secondary Stakeholders:

1. Regulators and laws
    a. General Data Protection Regulation GDPR
    b. Cookie Laws
    c. Alcohol Laws
2. Organizations focused on healthy alcohol consumption.

2.2 Requirement clarification

The requirements in this document follow the two templates *2.2.1 Template functional requirement* and *2.2.2 Template nonfunctional requirement*.

2.2.1 Template functional requirement

1. *Identifier:* An ID for the requirement in this project.

2. *Description:* A description of the requirement that shall guide the design and implementation of that requirement.

3. *Priority:* A rating of the importance of the requirement, it could be:
    a. *Essential:* The Requirement must be included in the application.
    b. *Useful:* The application will be less effective without the requirement.
    c. *Desirable:* Not a core in the application but the requirement will enhance the user experience of the application.

4. *Risk:* Project developers assessment of different risks in the requirement. A particular risk is graded with *low, medium* or *high* and could be one or more of the following risks:
    a. *Performance:* The requirement is considered to involve performance risk in the application.
    b. *Implementation:* The requirement is considered to be technically difficult to implement.
    c. *Schedule:* The requirement is considered to have a long implementation time and may not meet the deadlines or the implementation time of the requirement is hard to estimate.
    d. *Testability:* The requirement is considered hard to test.
    e. *Overall*: General risk assessment of the requirement.

2.2.2 Template nonfunctional requirement

1. *Identifier*, *Description*, *Priority*, and *Risk* follows the same description as the template for the functional requirement.

2. *Affected requirements:* What requirements this nonfunctional requirement affect

3. *Constraints:* What constraints this requirement has on the application.

2.3 General Requirements

Table 1: Classification and identification of Game Requirements.

Classes[1]	WD	UI	S	COM	NF
ID					
WD.UI.1	x	x			
WD.UI.2	x	x			
WD.UI.3	x	x			
WD.UI.4	x	x			
WD.S.1	x				
WD.UI.5	x	x			
WD.UI.6	x	x			
WD.UI.7	x	x			
WD.NF.1	x				x
WD.NF.2	x				x
WD.NF.3	x				x
WD.NF.4	x		x		x
WD.NF.5	x		x		x
WD.NF.6	x	x	x		x
WD.UI.8	x	x			
WD.UI.9	x	x			
WD.UI.10	x	x			
WD.UI.11	x	x			

Table 1: Classification and identification of Game Requirements.

Classes[1]	WD	UI	S	COM	NF
WD.COM.1	x			x	
WD.UI.12	x	x			
WD.UI.13	x	x			
WD.UI.14	x	x			
WD.UI.15	x	x			
WD.UI.16	x	x			
WD.UI.17	x	x			
WD.UI.18	x	x			

Classes[1]: WD = The Application WinnerDrinks, UI = User interface, S = Storage, persistent storage, session storage, COM = Communication with other systems, NF = Non functional requirement.

2.3.1 Enter game participants

Identifier: WD.UI.1

Description: A user shall be able to specify the game participant names that shall be included in the game. Only the valid game participants' names shall be added to the state of the game.

Game participant name rules:

1. A game participant’s name must be unique. Two unique names are names that contain the exact same number of characters in the same order transformed to lowercases or uppercases.
2. The length of a game participant’s name shall be between 1 and 10 letters.
3. A game participant’s name shall not contain any whitespaces.

Priority: Essential

Risk: Low overall risk

2.3.2 User error messages - Name input error message

Identifier: WD.UI.2

Description: If a user enters an invalid value in the name field, a user-friendly error message shall show up which lets the user go back and try again. Invalid values are 1) an empty value, 2) A value of more than 10 letters, and 3) An already taken name.

Content of a user friendly error message:

1. Reason of the error
2. The rules of a valid input

Priority: Desirable

Risk: Medium testability risk.

2.3.3 Pick game modules

Identifier: WD.UI.3

Description: A user shall be able to exclude or include game modules. Excluded game modules shall not appear in the game. The default setting shall be included and the minimum of included game modules shall be 2.

Priority: Essential

Risk: Low overall risk

2.3.4 Start the game

Identifier: WD.UI.4

Description: A user shall be able to start a game directly after he/she has entered a valid number of game participants. This means that all settings that a user can do except enter game participants must have a default setting.

Valid number of game participants:

1. Minimum number of game participants shall be equal to 2
2. Maximum number of game participants shall be equal to 10.

Default settings:

1. Every game-module is included except game modules that require more than 2 game participants, e.g game module *Back to Back*.
2. Every game participant is active, not paused.

Priority: Essential

Risk: Low overall risk.

2.3.5 Load game data

Identifier: WD.S.1

Description: When the game starts the game shall load all the necessary game event data for each game module, see requirement *2.3.9 Game modules reliability*.

Priority: Essential.

Risk: High Implementation risk, high performance risk.

2.3.6 Update game participants during a game

Identifier: WD.UI.5

Description: A user shall be able to update a game participant's name and the number of game participants during the game. Rules regarding the game's participants stated in requirement *2.3.1 Enter Game Participants* and *2.3.4 Start the Game* must be followed.

Priority: Useful

Risk: Low overall risk

2.3.7 Pause game participants during game

Identifier: WD.UI.6

Description: A user shall be able to pause game participants during an ongoing game, a paused game participant shall be marked as inactive. If less than two game participants are active then the game shall be paused.

Priority: Useful

Risk: Low overall risk

2.3.8 Participating game participants

Identifier: WD.UI.7

Description: Only the game participants that are active (not paused) shall be the game participants that can be part of a game event. The number of participating game participants is decided by each game event, see requirement *2.3.10 Game events' reliability*.

Priority: Useful

Risk: Low overall risk

2.3.9 Game modules reliability

Identifier: WD.NF.1

Description: The game in the application shall consist of reliable game modules.

Affected requirements:

1. *2.4 Module Party Requirements*
2. *2.5 Module Trivia (Multiple-choice) Requirements*
3. *2.6 Module Spin the Wheel Requirements*

Constraints

1. Each game module shall contain at least 20 unique text game events, e.g. Party or Trivia, **or** one random game event, e.g. Spin the Wheel.
2. If the game module contains text events, those events shall be stored in persistent memory that the application can access.
3. Each game event must fulfill requirements
    a. *2.3.11 Game events' reliability*
    b. *2.3.12 Game events usability*

Priority: Essential

Risk: Medium performance risk, Medium schedule risk.

2.3.10 Game modules extandability

Identifier: WD.NF.2

Description: The development of current and future game modules in the application must be able to take place independently of each other .

Affected requirements:

1. *2.4 Module Party Requirements*
2. *2.5 Module Trivia (Multiple-choice) Requirements*
3. *2.6 Module Spin the Wheel Requirements*
4. *2.7 Module Back to Back*

Constraints:

1. There must not be any coupling and dependencies between the game modules.

Priority: Essential

Risk: Medium implementation risk, medium testability risk.

2.3.11 Game events' reliability

Identifier: WD.NF.3

Description: A game event must be independent of other game events and shall be responsible for its own content, display, type and number of participating game participants that shall be in the game event.

Affected requirements:

1. Game modules and the game events

Constraints:

1. Allowed game event types (for an explanation of types, please see *1.5 Definitions and abbreviations*):
    a. Question
    b. Challenge
    c. Competition
    d. Luck (Random winner)
2. Each game event shall contain a predetermined number of game participants for each event. The number of game participants for each game event shall be between one and all.

Priority: Essential

Risk: Medium performance risk, medium testability risk.

2.3.12 Game events usability

Identifier: WD.NF.4

Description: A game event shall be easy to understand.

Affected Requirement:

1. *2.4 Module Party Requirements*, *2.4.1 Game Events*
2. *2.5 Module Trivia Requirements*, *2.5.1 Game Events*
3. *2.6 Module Spin the Wheel Requirements*, *2.6.1 Game Events*

Constraints text game events:

1. A game event's text content must not contain more than 60 words (≈ 20 seconds duration time Marc Brysbaert [1]). In the application text separated by a space shall be considered as a word.

Constraints animation game events:

1. A game animation event must not be longer than 7 seconds.

Priority: Essential

Risk: High implementation risk, high testability risk.

2.3.13 WinnerDrinks performance

Identifier: WD.NF.5

Description: The application shall perform well when a user interacts with the application, see the constraints for this requirement for more detailed information.

Constraints [1]

1. User interactions NOT connected to I/O operations: Response time <= 0.1 seconds.
2. User interactions connected to I/O operations and network requests during normal demands (= 1 visitor per second[2]): Response time <= 1 seconds.
3. User interactions connected to I/O operations and network requests during heavy demands (= 10 visitors per second[2]): Response time <= 10 seconds.

[1] The response time 0.1s, 1s, and 10 s for different user interactions is a useful agreed goal between the client and stakeholders working with the project. The data have been decided with the support of Jakob Nielsen [2].

[2] The normal and heavy demands are expected demands made by the client.

Affected requirements:

1. *2.3.5 Load game data*
2. *Requirement that is concerned with user interactions.*

Priority: Useful.

Risk: High testability risk, implementation risk.

2.3.14 WinnerDrinks portability

Identifier: WD.NF.6

Description: The application WinnerDrinks shall be fully functional and playable on a targeted list of web browsers, mobile devices and reading tablets.

Targeted list of web browsers:

1. Firefox 87-89:
2. Chrome 89-92
3. Edge 86-89
4. Android Browser 89
5. Chrome for android 89

Targeted list of mobile devices:

1. Iphones with iOS 13-14
2. Android phones with operating system versions 10 - 12.

Targeted list of reading tablets:

1. Ipads with iPadOS 13-14
2. Android tablets with operating system versions 10 - 12

Affected requirements: Requirement that is concerned with persistent storage direct or indirect, requirements that is concerned with the application GUI and application core functionality.

Constraints:

1. The application must be fully functional and playable in online conditions.
2. The application must be fully functional and playable in offline conditions.

Risk: High schedule and implementation risks, high testability risk.

2.3.15 Game event winner(s)

Identifier: WD.UI.8

Description: After every game event the winning participating game participant(s) must be selected, no winners can also be a possibility. If a game event winner can not be decided by the application (e.g the winner between two game participants in a real-life game of rock-paper-scissors) the user must be able to select the winner(s) in that event.

Priority: Essential

Risk: Low overall risk.

2.3.16 Transition to next game round

Identifier: WD.UI.9

Description: After the winner(s) have been selected, the current game round is considered over and this shall trigger a transition to the next game round. The next game round shall display following

1. The previous game round winner(s) names, see requirement *2.3.16 Winner announcement display*.
2. The previous game round winner(s) price, see requirement *2.3.17 Winner price display*.
3. The next game event, see requirement *2.3.20 Game event display*, shall be displayed.

Priority: Essential

Risk: Medium implementation risk, medium testability risk.

2.3.17 Game event display

Identifier: WD.UI.10

Description: After a game round has ended a new game event from one of the included game modules shall be displayed to the user. The game module shall be randomly selected and shall not be the last used game module.

Priority: Essential

Risk: Low overall risk.

2.3.18 Choose of new game event

Identifier: WD.UI.11

Description: A game event shall be available and randomly selected from a game modules pool of game events.

Priority: Essential

Risk: Low overall risk.

2.3.19 Winner announcement display

Identifier: WD.UI.12

Description: If the game is played in game mode standard, see requirement *2.3.22 Game variants*. Upon completion of a game event, the name of the winning participant shall be displayed on screen.

Priority: Essential

Risk: Low overall risk.

2.3.20 Winner price display

Identifier: WD.UI.13

Description: If the game is played in game mode standard, see requirement *2.3.22 Game variants*. Upon completion of a game event, the price to the winning participant shall be displayed on screen.

Priority: Essential

Risk: High volatile risk.

2.3.21 Game variants

Identifier: WD.UI.14

Description: When starting the application a user shall be able to choose between the two game variants *Scoreboard* and *Standard.* If the user chooses *Scoreboard* a scoreboard shall be displayed during the game, see requirement *2.3.21 Scoreboard display,* if the user chooses *Standard* the scoreboard will not be displayed.

Priority: Useful.

Risk: Medium volatile risk.

2.3.22 Scoreboard display

Identifier: WD.UI.15

Description: While a game is running using the game variant *Scoreboard* se requirement *1.5 Definitions and abbreviations*, a scoreboard with the game participants' names and scores shall be displayed. Upon completion of a game event, 1 point shall be added to the winning participants's total points.

Priority: Useful.

Risk: Medium volatile risk.

2.3.23 Skip game event

Identifier: WD.UI.16

Description: A user shall be able to skip a game event.

Priority: Desirable.

Risk: Low overall risk.

2.3.24 Game end

Identifier: WD.UI.17

Description: The game shall be infinite and has no end but a user shall be able to quit the game. This action shall erase current game session game participants.

Priority: Essential.

Risk: Low overall risk.

2.3.25 Game instructions

Identifier: WD.UI.18

Description: User instructions for the application, the game and various game modules shall be available on the application's start page.

Priority: Desirable.

Risk: Low overall risk.

2.4 Module Party Requirements

Table 2: Classification and identification of Module Party requirements.

Classes[1]	MP	UI	S	COM
MP.S.1	x		x	
MP.UI.1	x	x		
MP.UI.2	x	x		
MP.UI.3	x	x		

Classes[1]: MP = Module Party, UI = User interface, S = Storage, persistent storage, session storage, COM = Communication with other systems

2.4.1 Game events

Identifier: MP.S.1

Description: The game module shall contain at least 20 unique text game events of type question, challenges or competitions. These game events shall follow the constraints stated in requirement *2.3.9 Game modules reliability*, *2.3.11 Game events' reliability*, and *2.3.12 Game events usability*.

Priority: Essential

Risk: Medium implementation risk, medium testability risk.

2.4.2 Number of players

Identifier: MP.UI.1

Description: Two randomly picked game participants from the application's list of current active game participants shall be included in each party game event.

Priority: Essential

Risk: Low overall risk.

2.4.3 Game event display

Identifier: MP.UI.2

Description: A party game event shall display

1. The game event with the participating game participants included in the text.
2. Two buttons each containing a different participating game participant's name.

Priority: Essential

Risk: Low overall risk.

2.4.4 Choice of winner

Identifier: MP.UI.3

Description: A user shall be able to select the winner by clicking on one of the buttons containing a participating game participant's name, see requirement *2.4.3 Game event display*. This event shall trigger the transition to the next game round, see requirement *2.3.16 Transition to next game round*.

Priority: Essential

Risk: Low overall risk.

2.5 Module Trivia Requirements

Table 3: Classification and identification of Module Trivia requirements.

Classes[1]	MT	UI	S	COM
MT.1	x			
MT.UI.1	x	x		
MT.UI.2	x	x		
MT.UI.3	x	x		

Classes[1]: MT = Module Trivia, UI = User interface, S = Storage, persistent storage, session storage, COM = Communication with other systems.

2.5.1 Game events

Identifier: MT.S.1

Description: The game module shall contain at least 20 unique text game events of type questions These game events shall follow the constraints stated in requirement *2.3.9 Game modules reliability*, *2.3.11 Game events' reliability*, and *2.3.12 Game events usability*.

Priority: Essential

Risk: Medium implementation risk, medium testability risk.

2.5.2 Number of players

Identifier: MT.UI.1

Description: One randomly picked game participant from the application's list of current active game participants shall be included in each party game event.

Priority: Essential

Risk: Low overall risk

2.5.3 Game event display

Identifier: MT.UI.2

A trivia game event shall be displayed

1. The participating game participant
2. The game event
3. Multiple answer alternatives.

Priority: Essential

Risk: Low overall risk.

2.5.4 Choice of winner

Identifier: MP.UI.3 choice

Description: When the user clicks on one of the answer alternatives, see requirement *2.5.3 Game event display*, the game module shall decide if that was the correct answer. This event shall trigger the transition to the next game round, see requirement *2.3.16 Transition to next game round*.

Priority: Essential

Risk: Low overall risk.

2.6 Module Spin the Wheel Requirements

Table 4: Classification and identification of module Spin the Wheel requirements.

Classes[1]	MSW	UI	S	COM
MSW.1	x	x	x	
MSW.UI.1	x	x		
MSW.UI.2	x	x		
MSW.UI.3	x	x		

Classes[1]: MSW = Module Spin the Wheel, UI = User interface, S = Storage, persistent storage, session storage, COM = Communication with other systems

2.6.1 Game events

Identifier: MSW.1

Description: When the user clicks on the wheel in the game event, it runs an animation and a random participant is selected as the winner. This game event shall follow the constraints stated in requirement *2.3.9 Game modules reliability*, *2.3.11 Game events' reliability*, and *2.3.12 Game events usability*.

Priority: Essential

Risk: Medium implementation risk.

2.6.2 Game event display

Identifier: MSW.UI.1

Description: When an instance of the game module is started, a wheel with 4 sections shall be displayed for the user. Each section shall contain a randomly picked active game participant, the rules for this depends on the number of active game participants.

Rules of active game participants in wheel sections:

1. Active game participants >= 4: Each section shall contain a randomly picked game participant, max one section per game participant name.

2. Active game participant = 3: One game participant’s name shall be in two sections of the wheel.
3. Active game participant = 2: Both game participants' names shall be in two sections of the wheel.

Priority: Essential

Risk: Medium implementation risk.

2.6.3 Wheel spin

Identifier: MSW.UI.2

Description: By clicking on the wheel the user shall be able to make the wheel spin a random amount of degrees between 360 and 1080 (1-3 full rotations).

Priority: Useful

Risk: Medium implementation risk.

2.6.4 Choice of winner

Identifier: MSW.UI.3

Description: When the wheel has finished spinning, the winner is chosen from which name is at the top of the wheel. This event shall trigger the transition to the next game round, see requirement *2.3.16 Transition to next game round*.

Priority: Essential

Risk: Medium implementation risk.

2.7 Module Back to Back

Table 5: Classification and identification of module Spin the Wheel requirements.

Classes[1]	MSW	UI	S	COM
MBB.S.1	x	x	x	
MBB.UI.1	x	x		
MBB.UI.2	x	x		
MBB.UI.2	x	x	x	

Classes[1]: MSW = Module Spin the Wheel, UI = User interface, S = Storage, persistent storage, session storage, COM = Communication with other systems

2.7.1 Game events

Identifier: MBB.S.1

Description: The game module shall contain at least 20 unique text game events of types question, challenges and competitions. These game events shall follow the constraints stated in requirement *2.3.9 Game modules reliability*, *2.3.11 Game events' reliability*, and *2.3.12 Game events usability*.

Priority: Essential

Risk: Medium implementation risk, medium testability risk.

2.7.2 Number of players

Identifier: MBB.UI.1

Description: When the game module displays a new game event, 2 random game participants from the application's list of current active game participants' names shall be included in the content of the game event.

Priority: Essential

Risk: Low overall risk.

2.7.3 Game event display

Identifier: MBB.UI.2

A back to back game event shall display

1. The game event with the participating game participants included in the text.
2. One button displaying *Yes!*
3. One button displaying *No!*

Priority: Essential

Risk: Low overall risk.

2.7.4 Choice of winner

Identifier: MBB.UI.3

Description: A user shall be able to select if the participating game participants are winner(s) by clicking on the button displaying *Yes!* or losers by clicking on the button displaying *No!*, see requirement *2.7.3 Game event display*. This event shall trigger the transition to the next game round, see requirement *2.3.16 Transition to next game round*.

Priority: Essential

Risk: Low overall risk.

2.7 Requirements Analysis

We have performed a checklist-based analysis of the requirements continuously during the formation of the requirements and revised them accordingly. The requirements have been checked against the following list:

1. Premature design
2. Combined requirements
3. Unnecessary requirements
4. Use of non-standard hardware
5. Conformance with business goals
6. Requirements ambiguity
7. Requirements realism
8. Requirements testability

Some weaknesses in the requirements remain. These are specified below.

WD.UI.S.1 Enter game participants

Combined requirements: Could be 1) “specify player name” and 2) “add one player”.

WD.UI.2 User error messages

Ambiguity: Not specified exact scenario where the input is wrong.

WD.UI.3 Pick game modules

Combined requirements: Could be 1) “change included status”, 2) “if included, must not appear in game, and 3) “if not included, must not appear in game”.

WD.NF.1 Game modules reliability

Combined requirements: “Shall include modules” and “module must offer default setting” could be separate requirements.

2.7.1 Changes made based on the Requirements Analysis

Not all of the weaknesses have been addressed. After discussion during the project, we instead considered some of the remarks in the analysis not to be relevant. However, a number of changes and improvements were still made.

WD.UI.S.1 Enter game participants

Identifier changed to WD.UI.1. Reworded, so that it now concerns specifying a new player name, and validating input. Adding a name should be considered the same as adding a new player.

WD.UI.2 User error messages

The requirement now describes in detail what sort of input shall generate error messages. Therefore it is now more clear in exactly which scenarios this requirement is relevant.

WD.UI.3 Pick game modules

The description is slightly reworded and now includes a minimum number of included game modules.

WD.NF.1 Game modules reliability

This requirement now describes only that the application shall consist of reliable game modules. The functionality of included/excluded modules is handled in *2.3.3 Pick game modules*.

2.8 Requirements Validation

We have also performed a systematic validation of the requirements as a whole, during the formation of this document. The requirements have been checked against the following list:

1. Completeness
2. Consistency
3. Comprehensible
4. Ambiguous
5. Structure
6. Traceability
7. Standards

Comments on validation of the requirements

As for completeness, there may be missing requirements because of functionality that were not thought of at this point in time. This document is a living document that will be updated as requirements change or are added. For the time being, the requirements are complete.

The requirements are consistent with each other and are comprehensible to the reader. There may be some ambiguity as mentioned in the previous section. The requirements are structured so that general requirements have their own section, and module requirements are in respective sections. The requirements are named according to whether they mainly concern general function, UI, Storage, or Communication. In the case the requirements relate to other requirements, they are traceable as related requirements in the descriptions.

The document conforms to the defined standard set by the template given in the course (2DV609: D1 Requirements Specification).

2.8.1 Changes made based on the Requirements Validation

We have continuously during the development process changed, added, and removed requirements as needed to reflect new ideas and found limitations with the application or the development team. We have strived for completeness as the application works at this point in time.

Ambiguity in the requirements have been dealt with as we have improved the descriptions of the intended functionality and explained and elaborated more on certain concepts as needed and according to the feedback. For example, there are now two templates that explain how each requirement is structured and what is meant with risk, etc.

2.9 General test case proposals

The proposed test cases in this document follows the following template

Test case template:

1. *Test case name:* The title of each test case is equal to the name of the test case.
2. *Requirement:* A reference to the requirement in this document for which the test case is written.
3. *Requirement ID:* The ID for the requirement for which the test case is written.
4. *Test case ID:* The test case ID.
5. *Description:* The logical condition that the test case evaluates, the expected result is included.
6. *Preconditions:* List of conditions that must be true before this test case can start.
7. *Postconditions:* List of conditions that must be true when the test case ends
8. *Required data*: The type of data required for the test case.

General assumption test cases:

1. Game participants are implemented as a data type containing properties.
    a. Game participant name.
    b. Game participant status, active or not active.
    c. Game participant score.

2. Game module are implemented as a data type containing property
    a. Game module status, not active or active.

3. If a test case requires a list of game participants that does not specifically describe the number of game participants and the length of game participant names. Then this list is assumed to follow the game rules that apply to the number of game participants and their name.

4. If a test case requires a list of game modules that does not specifically describe the number of game modules. Then this list is assumed to follow the game rules that apply to the number of game modules.

2.9.1 Only valid game participants' name should be added to the state of the game

Requirement: *2.3.1 Enter game participants*

Requirement ID: WD.UI.1

Test case ID: T1.WD.UI.1

Description: When a user enters game participants' names then only the valid game participants' names, those that follow the rules in the requirement, should be added to the state of the game.

Precondition: No game participants' names in the state of the game.

Postcondition: The game should contain valid game participants names entered from the user.

Required data: A list of game participants containing invalid and valid names.

2.9.2 Game participants' name should be unique

Requirement: *2.3.1 Enter game participants*

Requirement ID: WD.UI.1

Test case ID: T2.WD.UI.1

Description: When a user enters game participants' names then only unique game participants names should be added to the state of the game. See the requirement under test for what is considered a unique game participants' names.

Precondition: No game participants' names in the state of the game.

Postcondition: The game should contain unique game participants' names according to the requirement.

Required data: A list of game participants containing valid but NOT unique names, eg “Joe” and “JOE”.

2.9.3 An invalid name input should display a correct error message.

Requirement: *2.3.2 User error messages - Name input error message*

Requirement ID: WD.UI.2

Test case ID: T1.WD.UI.2

Description: When a user enters a game participant name that does NOT conform to the rules of a game participant name. Then an error message containing the reason for the error and the rules of the input should be displayed for the user.

Eg an invalid game participant name could display:

"*That game participant's name is too long.*
A game participant name must be between 1-10 characters"

OBS! This test should be tested for every rule concerning a game participant's name.

Precondition: No game participants' names in the state of the game.

Postcondition: An error message containing the reason for the invalid input and the rules for that specific input.

Required data: A game participant with a name that contains more than the maximum number of valid characters in a game participant name.

2.9.4 A game session should contain at least two included game modules

Requirement: *2.3.3 Pick game modules*

Requirement ID: WD.UI.3

Test case ID: T1.WD.UI.3

Description: If exact two game modules are included the user should not be able to exclude more game modules.

Precondition: A game session with two included game modules.

Postcondition: A game session with two included game modules.

Required data: A list of two game modules and a list of game participants.

2.9.5 A game session should only contain included game modules

Requirement: *2.3.3 Pick game modules*

Requirement ID: WD.UI.3

Test case ID: T2.WD.UI.3

Description: When a specific number of game modules are set to included and the rest of the game modules are set to excluded. Then the game session should only contain the game modules that are set to be included.

Precondition: A game session with all game modules set to included.

Postcondition: A game session that only contains the included game modules.

Required data: A list of game modules and a list of game participants.

2.9.6 All but the game participants names should have default setting

Requirement: *2.3.4 Start the game*, ID: WD.UI.4

Requirement ID: WD.UI.4

Test case ID: T1.WD.UI.4

Description: If a user has entered a valid number of game participants' names and starts the game. Then the game should contain the default settings according to the requirement.

Precondition: No active game session.

Postcondition: A game session that contains all game modules and with all the game participants set to active (not paused).

Required data: A list of game modules and a list of game participants.

2.9.7 The game should contain the game events for each game module

Requirement: *2.3.5 Load game data* WD.S.1

Requirement ID: WD.S.1

Test case ID: T1.WD.S.1

Description: When the user starts a game then the game should contain at least 20 unique game events for each game module.

Precondition: No active game session.

Postcondition: A game session that contains at least 20 unique game events for each game module that require game event data.

Required data: A list of game modules and a list of game participants

2.9.8 The game should include the players' new names

Requirement: *2.3.6 Update game participants during a game*, ID: WD.UI.5

Requirement ID: WD.UI.5

Test case ID: T1.WD.UI.5

Description: Given that a user has started a game, when the user updates the game participants names then the application state should contain the new names for the game participants.

Precondition: Active game session with a certain number of game participants.

Postcondition: A game session that contains the new names of the game participants.

Required data: A list of game participants.

2.9.9 The game should be paused if only one active game participant

Requirement: *2.3.7 Pause game participants during game*,

Requirement ID: WD.UI.6

Test case ID: T1.WD.UI.6

Description: Given that a user plays the game and pauses all but one game participant. Then the game should be on pause.

Precondition: An game session with at least two game participants set to active (not paused).

Postcondition: A paused game session.

Required data: A list of game modules and a list of game participants.

2.9.10 Paused game participants should not be included in game events

Requirement: *2.3.8 Participating game participants*

Requirement ID: WD.UI.7

Test case ID: T1.WD.UI.7

Description: Given that a user has started a game and that a specific number of game participants are inactive, then when a new game event starts only the active game participants should be the ones that can be included in a game event.

Precondition: A game session.

Postcondition: A game session that contains game events with only the active (not paused) game participants.

Required data: A of game modules and a list of game participants.

2.9.11 Text game modules should contain at least 20 game events

Requirement: *2.3.9 Game modules reliability*

Requirement ID: WD.NF.1

Test case ID: T1.WD.NF.1

Description: Given that a game module is a game module that contains game events with text, eg Part and Trivia. Then that game module must store at least 20 game events for that game module.

Precondition: Access to game events that belong to the game module under test.

Postcondition: Game modules that store at least 20 game events.

Required data: None.

2.9.12 Text game modules should contain only unique game events

Requirement: *2.3.9 Game modules reliability*

Requirement ID: WD.NF.1

Test case ID: T2.WD.NF.1

Description: Given that a game module is a game module that contains game events with text, eg Part and Trivia. Then that game module must store only unique game events.

Precondition: Access to game events that belong to the game module under test.

Postcondition: Game modules that store only unique game events.

Required data: None

2.9.13 Game modules should not have any couplings

Requirement: *2.3.10 Game modules extensibility*

Requirement ID: WD.N2.2

Test case ID: T1.WD.NF.2

Description: Given that the application contains 3 game modules then there should not be any coupling between these game modules.

OBS! This test assumes that there is a tool that can be used to measure the coupling between the game modules.

Precondition: A specific number of game modules implemented in the application.

Postcondition: None couplings between game modules.

Required data: None

2.9.14 Participating game participants should be between one and all

Requirement: *2.3.11 Game events' reliability*

Requirement ID: WD.NF.3

Test case ID: T1.WD.NF.3

Description: Given that a user is playing the game with a valid number of active game participants. When a new game event is displayed then the participating game participants in that event should be between one and all of the active game participants.

Precondition: An active game session with a specific number of game participants set to active and the rest to inactive.

Postcondition: A game event that contains between one and all of the active game participants.

Required data: A list of game modules and a list of game participants.

2.9.15 Game events should not have more than 60 words

Requirement: *2.3.12 Game events usability*

Requirement ID: WD.NF.4

Test case ID: T1.WD.NF.4

Description: Given that a game module has game events containing text. Then that game module should only have game events containing text with less or equal than 60 words.

Precondition: Access to game events that belong to the game module under test.

Postcondition: Game events that contain less than 60 words.

Required data: None.

2.9.16 Game events animation should not be longer than 7 seconds

Requirement: *2.3.12 Game events usability*

Requirement ID: WD.NF.4

Test case ID: T2.WD.N4

Description: Given that a game module has game events that are animations. Then that animation should not take longer than 7 seconds.

Precondition: Access to the animation of the game event.

Postcondition: Game event animation that has been finished after 7 seconds from having started.

Required data: None.

2.9.17 User interactions NOT connected to I/O should not take more than 0.1s

Requirement: *2.3.13 WinnerDrinks performance*

Requirement ID: WD.NF.5

Test case ID: T1.WD.NF.5

Description: Given a user interaction that is not connected to I/O operations. Then the response from the application should not take longer than 0.1 seconds.

OBS! This test case assumes that there is a toll that can measure the user interactions

Precondition: Access to the application on the required platforms stated in the requirement *2.3.14 WinnerDrinks portability*.

Postcondition: User interactions that have response times less or equal to 0.1 seconds.

Required data: None.

2.9.18 User interactions connected to I/O should not take more than 1s

Requirement: *2.2.13 WinnerDrinks performance*

Requirement ID: WD.NF.5

Test case ID: T2.WD.NF.5

Description: Given a user interaction that is not connected to I/O operations. Then the response from the application should not take longer than 0.1 seconds.

OBS! This test case assumes that there is a toll that can measure the user interactions

Precondition: Access to the application on the required platforms stated in the requirement *2.3.14 WinnerDrinks portability*.

Postcondition: User interactions that have response times less or equal to 0.1 seconds.

Required data: None.

2.9.19 The application should be portable

Requirement: *2.3.14 WinnerDrinks portability*

Requirement ID: WD.NF.6

Test case ID: T1.WD.NF.6

Descriptions: The application should be fully functional and playable on the targeted list web browsers, mobile devices and reading tablets. This specifically means that the requirements in this document should apply when using the application on the various required platforms.

Precondition: Access to the application on the required platforms stated in the requirement *2.3.14 WinnerDrinks portability*.

Postcondition: A fully functional application on the required platforms.

Required data: None.

2.9.20 User should be able to select the game event winner(s)

Requirement: *2.2.15 Game event winner(s)*

Requirement ID: WD.UI.8

Test case ID: T1.WD.UI.8

Description: Given that a game event has a certain number of participating game participants and that the event is of the type that the user must decide the winner. When a user selects which game participants are the winner then only the participating game participants should be available for the user to select.

Precondition: A game event where the user must select the winner.

Postcondition: The winning game participants selected by the user should be between one and all of the participating game participants.

Required data: A list of game participants.

2.9.21 Application should be able to select the game event winner(s)

Requirement: *2.2.15 Game event winner(s)*

Requirement ID: WD.UI.8

Test case ID: T2.WD.UI.8

Description: Given that a game event is of the type that the application can decide the winner. When a winner is decided by the application then that winner should be one of the participating game participants.

Precondition: A game event where the application selects the winner.

Postcondition: The winning game participants selected by the application should be between one and all of the participating game participants.

Required data: A list of game participants.

2.9.22 Transition to next round should display winners, price and next game event

Requirement: *2.3.16 Transition to next game round*

Requirement ID: WD.UI.9

Test case ID: T1.WD.UI.9

Description: Given that a user is playing the game. When the winning game participants are selected then the transition to the next game event should be triggered and this should lead to display of a list of items described in the requirement under test.

Precondition: An active game session.

Postcondition: The display of the required list of items described in the requirement under test.

Required data: A list of game modules and a list of game participants.

2.9.23 Next game event should not be from the last used game module

Requirement: *2.3.17 Game event display*

Requirement ID: WD.UI.10

Test case ID: T1.WD.UI.10

Description: Given that a user is playing the game with two included game modules. Then the next game round should not include a game event from the previous used game module.

Precondition: An active game session.

Postcondition: A game round that does NOT include a game event from the game module that was used in the last round.

Required data: A list of game modules and a list of game participants.

2.9.24 If standard, game events winner names should be displayed on a screen

Requirement: *2.2.19 Winner announcement display*

Requirement ID: WD.UI.12

Test case ID: T1.WD.UI.12

Description: Given that a user is playing the game in game mode *Standard*. When the winning game participants are selected then the name(s) of those game participants should be displayed on the screen.

Precondition: A new game round started with a specified number of game participants.

Postcondition: The winning game participants displayed.

Required data: A list of game modules and a list of game events.

2.9.25 If standard, game event price should be displayed on the screen

Requirement: *2.2.20 Winner price display*

Requirement ID: WD.UI.13

Test case ID: T1.WD.UI.13

Description: Given that a user is playing the game in game mode *Standard*. When the winning game participants are selected then the price to those game participants should be displayed on the screen.

Precondition: A new game round started with a specified number of game participants.

Postcondition: The price to the winning game participants displayed.

Required data: A list of game modules and a list of game events.

2.9.26 User should be able to choose game variants

Requirement: *2.3.21 Game variants*

Requirement ID: WD.UI.14

Test case ID: T1.WD.UI.14

Description: Given that a user has started the application. Then the user should be able to select either game variant *Scoreboard* or *Standard*.

Precondition: The application has been started.

Postcondition: A game session using the selected game variant.

Required data: None.

2.9.27 If scoreboard, 1 point should be added to the winning game participants

Requirement: *2.3.22 Scoreboard display*

Requirement ID: WD.UI.15

Test case ID: T1.WD.UI.15

Description: Given that a user is playing the game using the game variant *Scoreboard*. When a game event is completed the winning game participants should be awarded 1 point to their total points.

Precondition: A game session started with a specified number of game participants all have zero points.

Postcondition: The winning game participants should have 1 point after the first game round.

Required data: A list of game modules and a list of game participants.

2.9.28 A user should be able to skip a game event

Requirement: *2.3.23 Skip game event*

Requirement ID: WD.UI.16

Test case ID: T1.WD.UI.16

Description: Given that a user is playing the game. When a new game event is displayed then the user should be able to skip that game event, this should lead to a new game event.

Precondition: A game session.

Postcondition: A new game event.

Required data: A list of game modules and a list of game participants.

2.9.29 Quit the game should erase game participants

Requirement: *2.3.24 Game end*

Requirement ID: WD.UI.17

Test case ID: T1.WD.UI.17

Description: Given that a user is playing the game. When a user quits the game then the current game participant should be erased. This specifically means that the properties names and score of the game participants should also be erased.

Precondition: A game session started.

Postcondition: No game participant should be in the application state.

Required data: A list of game modules and a list of game participants.

2.9.30 Game instruction should be available from the start page

Requirement: *2.3.25 Game instructions*

Requirement ID: WD.UI.18

Test case ID: T1.WD.UI.18

Description: Given that a user has started tha application. Then instructions about the application, the game, and the various game modules should be available for a user to view.

Precondition: The application displays the start page.

Postcondition: Displaying application instructions, game instruction, and instructions from the various game modules.

Required data: None.

2.10 Module Party test case proposals

Assumption module party test cases:

1. Party events can be rendered during tests without starting a game session.

2.10.1 Party game events should be reliable and usable

Requirement: *2.4.1 Game events*

Requirement ID: MP.S.1

Test case IDs:

1. T1.MP.S.1
2. T2.MP.S.1
3. T3.MP.S.1

Test case names:

1. Party game events should contain at least 20 game events.
2. Party game events should contain only unique game events.
3. Party game events should not have more than 60 words.

Description: This is a group of three test cases that test the reliability and usability of party game events. For more information see following general test cases for game events reliability and usability.

1. T1.WD.NF.1 *2.9.11 Text game modules should contain at least 20 game events*
2. T2.WD.NF.1 *2.9.12 Text game modules should contain only unique game events*
3. T1.WD.NF.4 *2.9.15 Game events should not have more than 60 words*

2.10.2 Party game events should contain 2 active game participants names

Requirement: *2.4.2 Number of players*

Requirement ID: MP.UI.1

Test case ID: T1.MP.UI.1

Description: Given a party game event has been rendered then that game event should contain 2 active game participants names.

Precondition: Access to party game events.

Postcondition: Rendered party game events each containing 2 active game participants.

Required data: A list of game participants.

2.10.3 Game module party should display a party game event

Requirement: *2.4.3 Game event display*

Requirement ID: MP.UI.2

Test case ID: T1.MP.UI.2

Description: Given a party game event has been rendered. When the event is displayed then the required list of items described in the requirement under test should be displayed.

Precondition: Access to party game events.

Postcondition: Rendered party events each displaying the required list of items described in the requirement under test.

Required data: A list of game participants.

2.10.4 Select a party event winner should trigger transition to next game round

Requirement: *2.4.4 Choice of winner*

Requirement ID: MP.UI.3

Test case ID: T1.MP.UI.3

Description: Given a party game event has been rendered. When a user clicks on one of the buttons containing a participating game participant this should trigger the transition to the next game round.

Precondition: Access to party game events.

Postcondition: The functionality that triggers a new game round should have been called.

Required data: A list of game participants.

2.11 Module Trivia test case proposals

Assumption module trivia test cases:

1. Trivia events can be rendered during tests without starting a game session.

2.11.1 Trivia game events should be reliable and usable

Requirement: *2.5.1 Game events*

Requirement ID: MT.S.1

Test case IDs:

1. T1.MT.S.1
2. T2.MT.S.1
3. T3.MT.S.1

Test case names:

1. Trivia game events should contain at least 20 game events.
2. Trivia game events should contain only unique game events.
3. Trivia game events should not have more than 60 words.

Description: This is a group of three test cases that test the reliability and usability of trivia game events. For more information see following general test cases for game events reliability and usability.

1. T1.WD.NF.1 *2.9.11 Text game modules should contain at least 20 game events*
2. T2.WD.NF.1 *2.9.12 Text game modules should contain only unique game events*
3. T1.WD.NF.4 *2.9.15 Game events should not have more than 60 words*

2.11.2 Trivia game events should contain 1 active game participant's name

Requirement: *2.5.2 Number of players*

Requirement ID: MT.UI.1

Test case ID: T1.MT.UI.1

Description: Given a trivia game event has been rendered then that game event should contain 1 active game participant's names.

Precondition: Access to trivia game events.

Postcondition: Rendered trivia game events each containing 1 active game participant.

Required data: A list of game participants.

2.11.3 Game module trivia should display a trivia game event

Requirement: *2.5.3 Game event display*

Requirement ID: MT.UI.2

Test case ID: T1.MT.UI.2

Description: Given a trivia game event has been rendered. When the event is displayed then the required list of items described in the requirement under test should be displayed.

Precondition: Access to trivia game events.

Postcondition: Rendered trivia events each displaying the required list of items described in the requirement under test.

Required data: A list of game participants.

2.10.4 Select a answer alternative should trigger transition to next game round

Requirement: *2.5.4 Choice of winner*

Requirement ID: MT.UI.3

Test case ID: T1.MT.UI.3

Description: Given a trivia game event has been rendered. When a user clicks on one of the buttons containing an answer alternative this should trigger the transition to the next game round.

Precondition: Access to trivia game events.

Postcondition: The functionality that triggers a new game round should have been called.

Required data: A list of game participants.

2.12 Module Spin the Wheel test case proposals

Assumption module spin the wheel test cases:

1. Spin the wheel events can be rendered during tests without starting a game session.

2.12.1 Spin the Wheel animation should not be longer than 7 seconds

Requirement: *2.6.1 Game events*

Requirement ID: MSW.S.1

Test case ID: T1.MSW.S.1

Description: See test cases

T2.WD.N4 *2.9.16 Game events animation should not be longer than 7 seconds*

2.12.2 Wheel section should contain names according to the rules

Requirement: *2.6.2 Game event display*

Requirement ID: MSW.UI.1

Test case ID:

1. T1.MSW.UI.1
2. T2.MSW.UI.1
3. T3.MSW.UI.1
4. T4.MSW.UI.1

Test case name:

1. Wheel section content should follow rules, 2 active game participants.
2. Wheel section content should follow rules, 3 active game participants.
3. Wheel section content should follow rules, 4-10 active game participants.

Description: Given a *Spin the Wheel* game event has been rendered. When a new game event starts the wheel sections should contain active game participants according to the rules described in the requirement under test.

Precondition: Access to render a spin the wheel game event.

Postcodition: A spin the wheel displaying a wheel according to the rules described in the requirement under test.

Required data: A list of game participants.

2.12.3 A click on the wheel should make the wheel spin 360-1080 degrees

Requirement: *2.6.3 Wheel spin*

Requirement ID: MSW.UI.2

Test case ID: T1.MSW.UI.2

Description: Given that a user has clicked on the wheel. Then the wheel should rotate between 360 and 1080 degrees (1-3 full rotations).

Precondition: A rendered spin the wheel game event.

Postcondition: The wheel rotated between 360 and 1080 degrees.

Required data: A list of game participants.

2.12.4 The wheel should point to the winner

Requirement: *2.6.4 Choice of winner*

Requirement ID: MSW.UI.3

Test case ID: T1.MSW.UI.3

Description: Given that the wheel has finished spinning. Then the name the arrow is pointed to should be the name of the winning game participant.

Precondition: A rendered spin the wheel game event.

Postcondition: A winning game participant.

Required data: A list of game participants.

2.13 Module BackToBack test case proposals

Assumption module back to back test cases:

1. Back to back events can be rendered during tests without starting a game session.

2.13.1 Back to back game events should be reliable and usable

Requirement: *2.7.1 Game events*

Requirement ID: MBB.S.1

Test case IDs:

1. T1.MBB.S.1
2. T1.MBB.S.1
3. T1.MBB.S.1

Test case name:

1. Back to back game events should contain at least 20 game events.
2. Back to back game events should contain only unique game events.
3. Back to back game events should not have more than 60 words.

Description: This is a group of three test cases that test the reliability and usability of trivia game events. For more information see following general test cases for game events reliability and usability.

1. T1.WD.NF.1 *2.9.11 Text game modules should contain at least 20 game events*
2. T2.WD.NF.1 *2.9.12 Text game modules should contain only unique game events*
3. T1.WD.NF.4 *2.9.15 Game events should not have more than 60 words*

2.13.2 Back to back game events should contain 2 active game participants names

Requirement: *2.7.2 Number of players*

Requirement ID: MBB.UI.1

Test case ID: T1.MBB.UI.1

Description: Given a back to back game event has been rendered then that game event should contain 2 active game participants names.

Precondition: Access to back to back game events.

Postcondition: Rendered back to back game events each containing 2 active game participants.

Required data: A list of game participants.

2.13.3 Game module back to back should display a back to back game event

Requirement: *2.7.3 Game event display*

Requirement ID: MBB.UI.2

Test case ID: T1.MBB.UI.2

Description: Given a back to back game event has been rendered. When the event is displayed then the required list of items described in the requirement under test should be displayed.

Precondition: Access to back to back game events.

Postcondition: Rendered back to back events each displaying the required list of items described in the requirement under test.

Required data: A list of game participants.

2.13.4 Button "*Yes*" should trigger transition to the next game round

Requirement: *2.7.4 Choice of winner*

Requirement ID: MBB.UI.3

Test case ID: T1.MBB.UI.3

Description: Given a back to back game event. When a user clicks on the button containing text "Yes!" it should trigger the transition to the next game round.

Precondition: Access to back to back game events.

Postcondition: The functionality that triggers a new game round should have been called.

Required data: A list of game participants.

2.13.5 Button "*No*" should trigger transition to next game round

Requirement: *2.7.4 Choice of winner*

Requirement ID: MBB.UI.3

Test case ID: T2.MBB.UI.3

Description: Given a back to back game event. When a user clicks on the button containing text “No” it should trigger the transition to the next game round.

Precondition: Access to back to back game events.

Postcondition: The functionality that triggers a new game round should have been called.

Required data: A list of game participants.

3. System Interfaces

3.1 User Interfaces

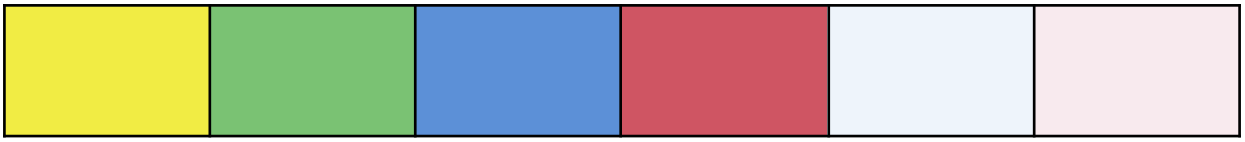
The User Interface that will be implemented is a Graphical User Interface, which accepts input to the software in form of text as well as events when the user presses a button. The Interface will be easy to use and understand. The language in the application will be english to increase the number of people that can use it.

3.1.1 Look & Feel

Colors

The (assumed) client has provided us with a color-palette that will be used for the software's coloring of the background, buttons, boxes, etc. The following listed hex colors has been provided:

- Yellow: #F1EC44
- Green: #7AC273
- Blue: #5C90D7
- Red: #CF5563
- Light blue: #EEF4FB
- Light red: #F8EAEE



Font

In the text nodes in the application and in the Graphical User Interface, one of the following two fonts will be used: "Segoe UI" for buttons and menus, and "Times New Roman" for the title, scoreboard, module text, and other text nodes that are not buttons or menus.

Navigation/Interaction

The client has provided us with a simple "one way to go" description on how the users shall interact with the application. By using the pointer and interacting with the application, the users shall be encouraged to follow a one-way path from the beginning to the end. However, other navigation options such as "back to start page" are allowed if this is deemed necessary to make the application more user friendly.

The software will consist of a single page that makes one request to the server that responds with the content. After that, each interaction will dynamically re-render the page and replace the content with the content desired rather than making new requests for each interaction.

3.1.2 Layout and Navigation Requirements

The application shall be simple and shall not have too many components on the screen. The application shall have an easily understandable icon, in the top left corner, used to open the sidebar consisting of possible settings to update.

Start page interface content

1. Header displaying Winner Drinks.
2. Start button.
3. Button to change game mode *Standard* or *Scoreboard.*
4. Button to add game participants.
5. Text-input to enter game participants' names.
6. Settings menu.
7. Footer containing links to instruction and alcoholic recommendations.

Spin-the-wheel GUI
The Spin the Wheel GUI will have a wheel with 4 sections with a unique color to simply separate the different areas. When playing the game each section of the wheel shall contain a game participant's name.

Trivia GUI
The trivia GUI will consist of a text field where the software randomly generates and displays one question. Underneath the question, two or more buttons with answers will be displayed that the user can guess the correct answer from. This button group will be aligned according to 2x2 if there are four buttons. Otherwise in columns.

Party challenges GUI
The GUI of the party challenge module will consist of a text area with a game event containing the participating game participants and below a button group each with each containing the selected game participant's name in it.

Back to Back GUI
The GUI of the party challenge module will consist of a text area with a game event containing the participating game participants and below 2 buttons, one button to be used if the game participants succeed or on if they don't.

Settings GUI
The settings interface will consist of a sidebar from the right with grouped checkboxes for the different game-modules as well as each participant's name and buttons that can paus, re-start, and delete a game participant.

3.1.3 Consistency

Gear icon
It is usual and predictable that a gear-icon opens the settings on a web application or mobile app. This will also be the use for our web-application, where the gear-icon indicates that the user can change the settings.

Checkboxes & Radio Buttons
It is predictable for a user that a checkbox is a type of input which can be checked, or unchecked. Checked indicates that the statement is enabled, and an unchecked checkbox indicates on disabled.

Button text
Consistency such as right choice of text in the buttons in the application will help the users to predict what will happen next.

3.1.4 User Personalization & Customization Requirements

There are no requirements based on users attributes except for the dynamically generated questions and challenges which are produced according to the participants names. Example: “Fred challenges Nina on a rap-battle”. Where Fred and Nina are some of the names of the participants and the challenge is generated according to the names.

Beyond that, the only customization requirement needed and stated is that the user should be able to change their settings, such as what type of game modules (s)he wants to play and what type of questions that should be displayed. The application must adjust to this and ignore the game-modules not wanted by the user.

3.2 Interfaces to External Systems or Devices

3.2.1 Software Interfaces

Since the application is a web-application, it will need support for a web-browser to work and perform.

3.2.2 Hardware Interfaces

The application works on mobile devices, tablets, and laptops.

The software does also work on desktops, but an external screen, mouse and keyboard will then be required.

3.2.3 Communications Interfaces

The application is using the HTTPS-protocol and communicates over the internet.

4. Domain Rules

4.1 General rules

4.1.1 Alcoholic recommendations

If playing WinnerDrinks then information about alcoholic recommendations shall be available for the game participants.

“Warning!
Drink responsibly.
Never drink and drive.
Never drink if you are underage.
Never drink if you are pregnant”.

4.2 Game Rules

4.2.1 Number of game participants

If a user is to determine the number of game participants

- ➔ Then inform the user that the number of game participants must be between 2 and 10 players and that the identification of a game participant must be unique.

4.2.2 Starting the game

If a user starts the game then

- ➔ Shuffle the game events.
- ➔ Inform the user about the picked game modules names and the general goals of the games.

4.2.3 Playing the game

If a user plays in game mode *Scoreboard* and have finished a game event then

- ➔ Inform the user the total score of the game participants.
- ➔ Start a new game event.

If a user plays in game mode *Standard* and have finished a game event then

- ➔ Inform the game participants about the prizes from the just finished game event.
- ➔ Start a new game event.

5. System Constraints

Under this section, you will find the constraints that the client has on the application and the project.

5.1 List of project constraints

1. The project Winner Drinks must be documented in English

2. The project documentation, the Project Proposal, The Requirement Specification, and the Design Document shall be available for the client.

3. The source code shall be available at https://github.com/2dv609/WinnerDrinks.

5.2 Application constraints

1. The language of the application must be in english.

2. The application shall be released 2021-05-30

6 Use-Cases

Use Case 1 - Start the game

Description: A user shall be able to enter the game participants name and then start the game.

Actors: A User, Game participants.

Pre-condition: None.

Post-condition: The game has started.

Basic flow:

1. The users starts the application
2. The application displays the start page including an input field for a game participant name.
3. The user enters one game participant name and presses enter or the “add”-button.
4. The application adds the entered name to the game participants' names and repeats point 2.
5. The user clicks on the play button.
6. The application starts the game, see use case 3 *Play the game*.

Alternative flow: See use case 14 *Error handling of game participants* alternative flows 1-4.

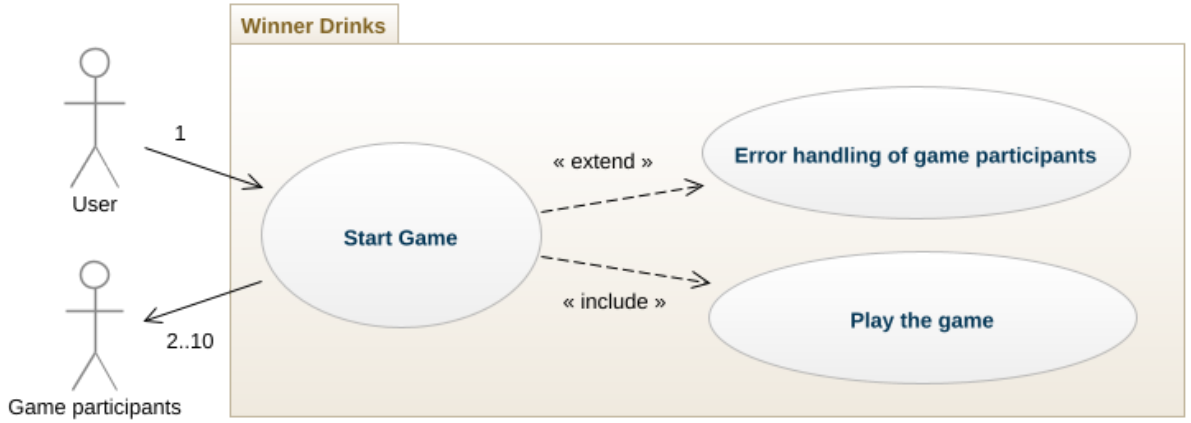


Figure 1: Use case diagram Start the game

Use Case 2 - Choose game module

Description: A user should be able to disable the by default included game modules.

Actors: A User.

Pre-condition: The application has been started.

Post-condition: The inclusion status of the game modules is updated.

Basic flow:

1. The application displays the start page.
2. The user clicks on the setting menu.
3. The application displays the game modules implemented in the application and a checked checkbox if the game module is included or unchecked if excluded.
4. The user selects exclude or include for one or several game modules and then close the settings menu.

Alternative flow:

4.1 The user returns to the previous page without updating the game modules.

1. The user closes the settings menu.

4.2 Only two game modules are included

1. The user tries to uncheck one of the two included game modules.
2. The application does nothing.

Figure 2: Use case diagram Choose game module

Use case 3 - Play the game

Description: A user shall be able to play the game.

Actors: A User, Game participants.

Pre-condition: The game has been started, see use case 1 *Start the game*.

Post condition: The application displays the start page.

Basic flow:

1. The application displays
    a. A random game event from a random included game module with between 1 and all of the game participants included.
    b. The previous game rounds game participants winners and prizes.
2. The user plays the random game event from a random included game module, see use case 4-7, *Spin the wheel*, *Trivia*, *Party*, or *Back to Back*.
3. The application repeats point 1.

Alternative flow:

1b.1 First round in the game or the previous round have been skipped, see use case 13 *Skip a game round*.

1. The application does NOT display prizes or winners, only the next game event is displayed.

1b.2 The user have started the game with game mode *Scoreboard*

1. The application displays the scoreboard, see requirement *2.3.22 Scoreboard display*

2.1 The user quit the game, see use case 12 *Quit the game.*

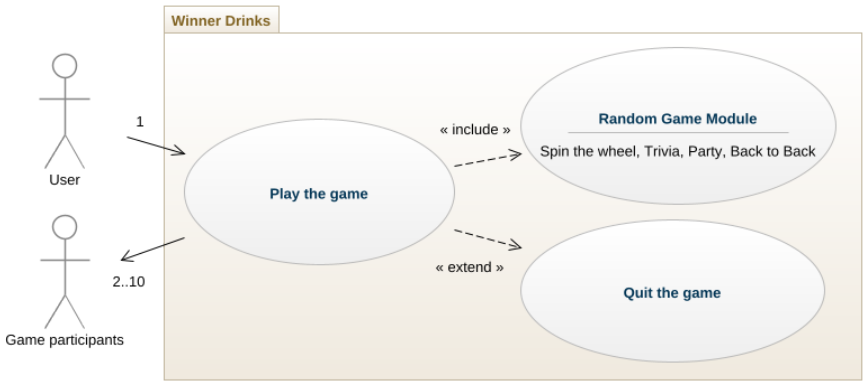


Figure 3: Use case diagram Play the game

Use Case 4 - Spin the wheel

Description: Play a game round of type *Spin the wheel*.

Actors: A User, Game participants.

Pre-condition: The game has been started, see use case 1. *Start the game* and 3. *Play the game.*

Post-condition: The next game round or the start page.

Basic flow:

1. The application displays a spin the wheel game event with the game participant names included in the wheel according to requirement rules, see requirement *2.6.2 Game event display*
2. The user clicks on the wheel
3. The application
    a. Spins the wheel and selects a winner.
    b. Make a transition to the next game round, see requirement *2.3.16 Transition to next game round* and use case 3 *Play the game.*

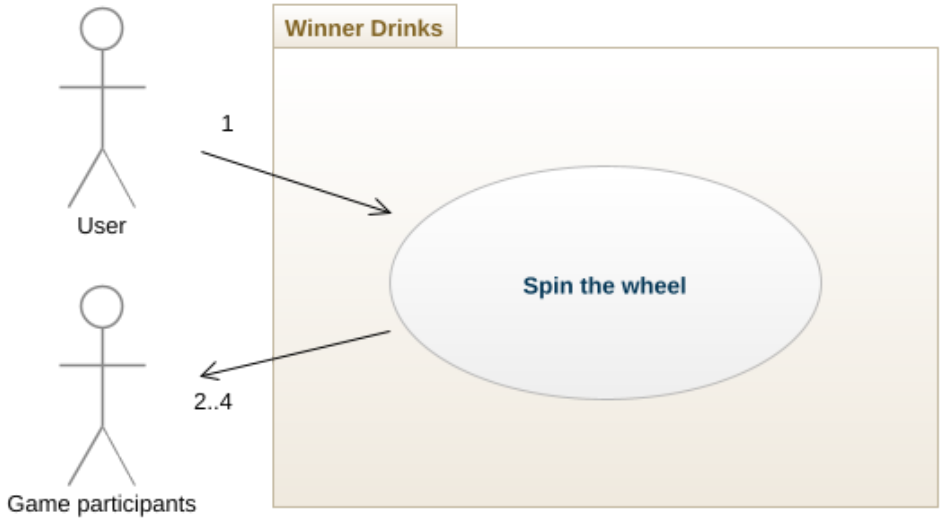


Figure 4: Use case diagram Spin the wheel

Use Case 5 - Trivia

Description: Play a game round of type *Trivia*.

Actors: A User, Game participants.

Pre-condition: The game has been started, see use case 1. *Start the game* and 3. *Play the game.*

Post-condition: The next game round or the start page.

Basic flow:

1. The application displays a trivia game event, a question, with one included active game participant and the possible answers.
2. The user clicks on a possible answer.
3. The application makes a transition to the next game round, see requirement *2.3.16 Transition to next game round* and use case 3 *Play the game*.

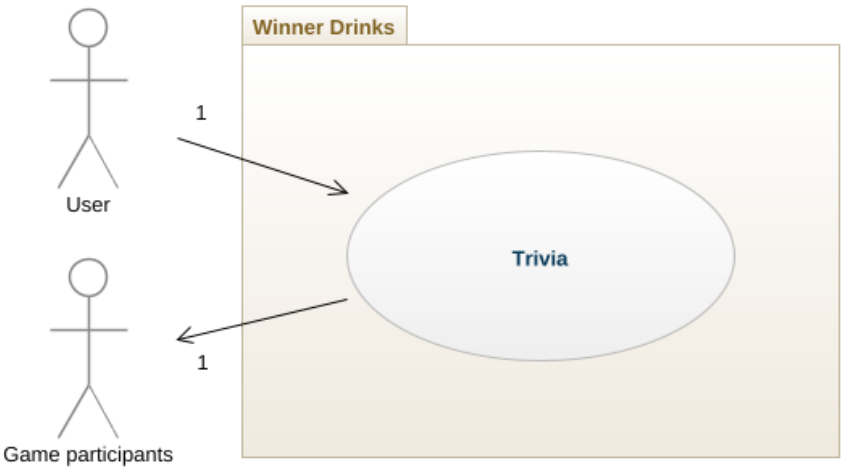


Figure 5: Use case diagram Trivia

Use Case 6 - Party

Description: Play a game round of type *Party*.

Actors: A User, Game participants.

Pre-condition: The game has been started, see use case 1. *Start the game* and 3. *Play the game.*

Post-condition: The next game round or the start page.

Basic flow:

1. The application displays a party game event, e.g. a challenge, with two active game participants included and two buttons each with a name from the included participants.
2. The user clicks on off the buttons containing the succeeded participants off the challenge.
3. The application makes a transition to the next game round, see requirement *2.3.16 Transition to next game round* and use case 3 *Play the game*.

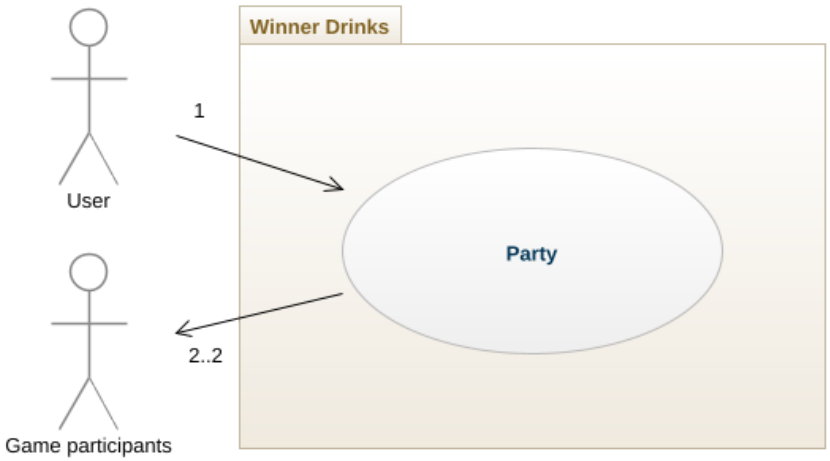


Figure 6: Use case diagram Party

Use Case 7 - Back to Back

Description: Play a game round of type *Party*.

Actors: A User, Game participants

Pre-condition: The game has been started, see use case 1. *Start the game* and 3. *Play the game.*

Post-condition: The next game round or the start page.

Basic flow:

1. The application displays a back to back game event with two active game participants included and two buttons with content *Yes!* and *No!*.
2. The user clicks on the player that succeeded the challenge.
3. The application makes a transition to the next game round, see requirement *2.3.16 Transition to next game round* and use case 3 *Play the game*.

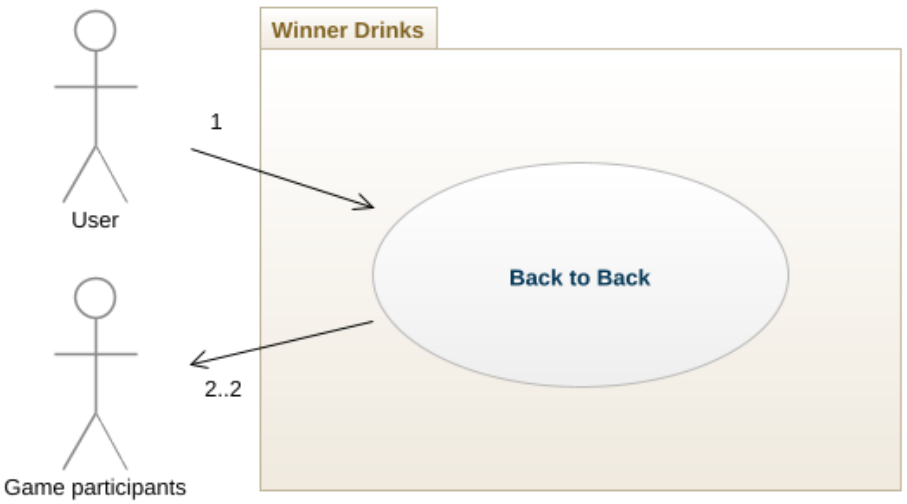


Figure 7: Use case diagram Back to Back

Use Case 8 - Add a game participant in-game

Description: While in a running game, a user shall be able to add game participants to the game-session.

Actors: A User, a Game participant.

Pre-condition: A game session and a random game event is displayed, see use case 1. *Start the game.*

Post-condition: A new game participant has been added, if basic flow, to the game-session and the same game event as before the use case is displayed.

Basic flow:

1. The application has been started and displays one random game event.
2. The user opens the settings menu.
3. The application displays a list of the current game participants and an input field to add new game participants.
4. The user enters one new game participant's name and clicks add.

Alternative flow:

3.1 The user chooses to quit the game, see use case 12 *Quit the game*.
4.1 The user makes some invalid input when entering a new game participant name, see use case 14 *Error handling of game participants* alternative flows 1-4.

Figure 8: Use case diagram Add a game participant in-game

Use Case 9 - Delete a game participant in-game

Description: While in game, a user shall be able to remove a game participant in the game-session.

Actors: A User, A Game participant

Pre-condition: The game has been started, see use case 1. *Start the game.*

Post-condition: A game participant has been removed from the game-session.

Basic flow:

1. The application has been started and displays one random game event.
2. The user opens the settings menu.
3. The application displays a list of the current game participants and a form with input.
4. The user clicks on the delete button for a specific game participant.
5. The application asks for confirmation.
6. The user confirms the deletion of the selected game participant.
7. The application deletes the elected game participant.

Alternative flow:

3.1 The user chooses to quit the game, see use case 12 *Quit the game*.

4.1 The user makes some invalid choice when deleting a game participant, see use case 14 *Error handling of game participants* alternative flow 5.

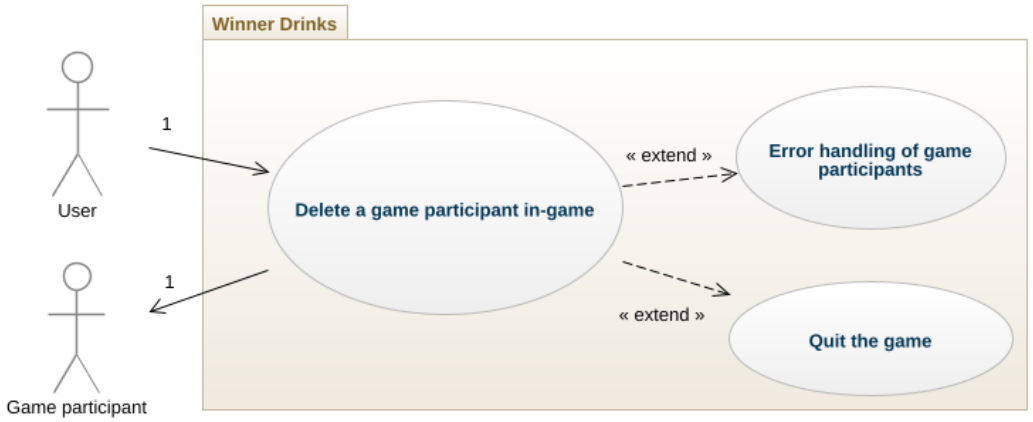


Figure 9: Use case diagram Delete a game participant in-game

Use Case 10 - Change pause status for a game participant in-game

Description: A user shall be able to change pause status for a game participant in game.

Actors: A User, a Game participant.

Pre-condition: The game has started, see use case 1. *Start the game.*

Post-condition: A game participant pause status has been changed.

Basic flow:

1. The application has been started and displays one random game event.
2. The user opens the settings menu.
3. The application displays a list of the current game participants.
4. The user changes pause status for a game participant in the list, and closes the settings menu.

Alternative flow:

3.1 The user chooses to quit the game, see use case 13 *Quit the game*.
4.1 The user makes some invalid choice when pausing a game participant, see use case 14 *Error handling of game participants* alternative flow 5.

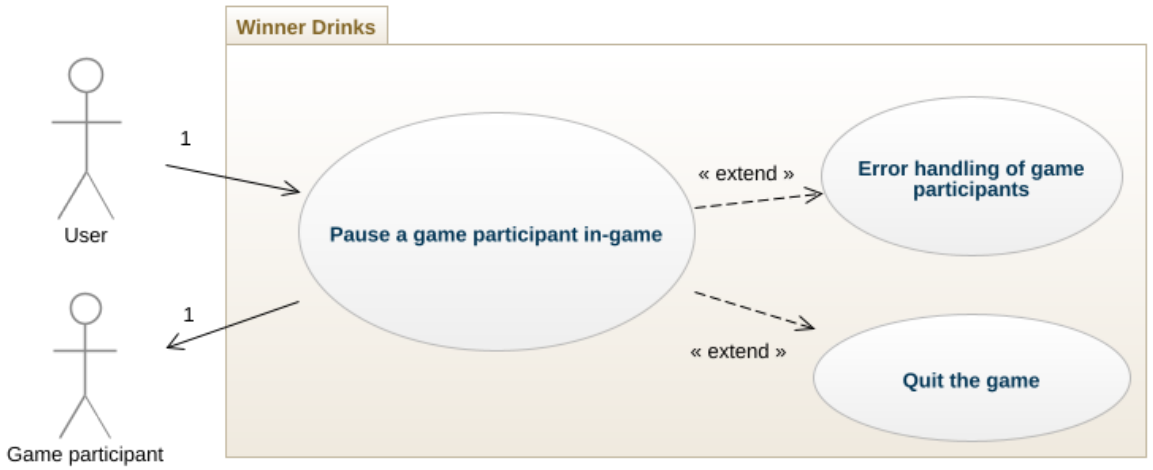


Figure 10: Use case diagram Pause a game participant in-game

Use Case 12 - Choose game mode

Description: A user shall be able to choose from the game modes *Standard* and *Scoreboard*.

Actors: A User.

Pre-condition: The application has been started

Post-condition: The mode is changed to reflect the user's choice.

Basic flow:

1. The application displays modes available to choose.
2. The user presses one of the available options *Standard* or *Scoreboard*.
3. The application confirms and updates the game mode to the selected mode.

Figure 11: Use case diagram Choose game mode

Use case 13 - Quit the game

Description: A user shall be able to quit the infinite game.

Actors: A User, Game participants.

Pre-condition: The application has been started

Post-condition: The application displays the start page and all game participants including the game participants score and names have been erased.

Basic flow:

1. The application displays a random game event.
2. The user opens the settings menu and chooses to reset/quit the game.
3. The application resets the game, erases the game participants, and redirects the user to the applications start page.

Figure 12: Use case diagram Quit the game

Use case 14 - Skip a game round

Description: A user shall be able to skip a game round.

Actors: A User.

Pre-condition: The application has been started and displays a random game event.

Post-condition: The application displays a new random game event with no winners and prizes from the previous game round.

Basic flow

1. The application displays a random game event from a random included game module.
2. The user chooses to skip the game event.
3. The application displays a new random game event from an included game module that is not the previous game module with no winners and prizes from the previous game round.

Figure 13: Use case diagram Skip a game round.

Use case 15 - Error handling of game participants

Description: A user enters a game participant name incorrect or the number of game participants are not valid for a game session to start. This use case is an extension to use cases handling input from a user concerning game participants.

Actors: A User.

Pre-condition: A user has entered a game participant name incorrectly or the number of game participants has not been valid for a game session to start.

Post-condition: An informative error message displayed to the user.

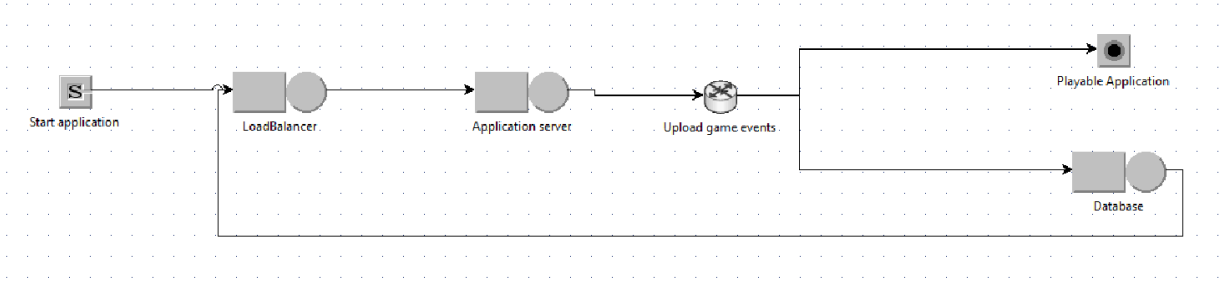
Alternative Flow:

1. The user skips to enter a name and presses the “add”-button.
    1. The application displays an error message that says that a name must be between 1-10 letters.
2. The user enters a name that contains spaces and then presses the “add”-button.
    1. The application displays an error message that says that a name must not contain any spaces.
3. 10 game participants' names are added to the game and the user tries to add one more.
    1. The application displays an error message that says max 10 game participants are allowed.
4. 0 or 1 game participants are added to the game and the user clicks on the play button on the start page.
    1. The application displays an error message that says 2-10 game participants must be added.
5. The user pauses or deletes all or all but one game participants during a game session.
    1. The application displays an error message that says too many game participants are paused.

Figure 14: Use case diagram Error handling of game participants

7. Performance modeling and analysis

Figure 15: Queueing network model over the system winner drinks



The application will be designed so that all data is retrieved when the application starts. To optimize the performance with the minimum number of resources when deploying the application we need to know how many instances we need for the database and the application server. To get this information the clients expected normal demands and heavy demands of the system is used, see constraints in requirement *2.3.13 WinnerDrinks performance*.

Assumptions

- It has been assumed that the utilization for the database and application server shall not be over 50 % to not risk being overloaded.
- During this analysis it is assumed that the load balancer will have zero response time.
- Service time database requests, based from previous knowledge: 120 ms
- Service time Application server requests, based from previous knowledge: 120 ms

7.1 Result

The result from the analysis

1. Optimized number of instances of the database normal conditions: 1
2. Optimized number of instances of the database heavy conditions: 8
3. Optimized number of instances of the Application server normal conditions: 1
4. Optimized number of instances of the application server normal conditions: 10

The result is based on assumption, from previous knowledge, for the database requests service time and for the server application requests service time. When the system is deployed actually data could be collected to get more accurate results.

Table 6: Performance data from the analysis of the system during expected normal- and heavy demands

Diagram Components	**Throughput normal demand (req/s)**	**Throughput heavy demand (req/s)**	**Service Time (ms)**
Application server Serve static files and requests to get game events.	4	40	120
Database Store game events	3	30	120
Load balancer	-	-	-
Start application A user starts the application	-	-	-
Playable Application All game events to the application have been uploaded.	-	-	-
WinnerDrinks The whole system	1 requests/s	10 req/s	-

7.2 Calculations

Performance laws used in calculations

- ➔ Utilization law: Utilization[component] = throughput[component] * service time[component] / instances[components]
- ➔ Forced Flow law: throughput[component] = throughput * visitor[component]

Visitor Database and Application server

Visitor Application server: Va = [V] * the number of implemented game modules that require persistent game events plus one for the static sides.

Visitor Database: Vd = [V] * the number of implemented game modules that require persistent game events.

7.2.1 Normal conditions

- Implemented game modules that require persistent game events: N = 3
- Visitor system: Vs = 1
- Throughput System: Xs = 1 req/s

- Optimized utilization Database: Ud = 0.5
- Visitor database: Vd = Vs * N = 3
- Throughput Database: Xd = Xs * Vd = 3 req/s, Forced Flow law
- Service time Database: Sd = 0.120 s
- Optimized number of instances Databases, must be an integer: Cd = ?

- Optimized utilization Application server: Ua = 0.5
- Visitor Application server: Va = Vs * N + 1 = 4
- Throughput Application server: Xa = Xs * Va = 4 req/s, Forced Flow law
- Service time Application server: Sa = 0.120 s
- Optimized number of instances Application server, must be an integer: Ca = ?

Utilization law =>

Ud = Xd * Sd / Cd =>
Cd = Xd * Sd / Ud = 3 * 0,120 / 0.5 = 0.72 =>
Cd = 1

Ua = Xa * Sa / Cd =>

Ca = Xa * Sa / Ua = 4 * 0,120 / 0.5 = 0.96 =>
Ca = 1

7.2.2 Heavy conditions

- Implemented game modules that require persistent game events: N = 3
- Visitor system: Vs = 10
- Throughput System: Xs = 10 req/s

- Optimized utilization Database: Ud = 0.5
- Visitor database: Vd = Vs * N = 30
- Throughput Database: Xd = Xs * Vd = 30 req/s
- Service time Database: Sd = 0.120 s
- Optimized number of instances Databases, must be an integer: Cd = ?

- Optimized utilization Application server: Ua = 0.5
- Visitor Application server: Va = Vs * N + 1 = 40
- Throughput Application server: Xa = Xs * Va = 40 req/s
- Service time Application server: Sa = 0.120 s
- Optimized number of instances Application server, must be an integer: Ca = ?

Utilization law =>

Ud = Xd * Sd / Cd =>
Cd = Xd * Sd / Ud = 30 * 0,120 / 0.5 = 7.2 =>
Cd = 8

Ua = Xa * Sa / Cd =>
Ca = Xa * Sa / Ua = 40 * 0,120 / 0.5 = 9.6 =>
Ca = 10

8 Requirement Specification changelog

This section describes major changes that have been made to this document's requirements between different versions of this document. Since we work with an agile approach in this project to be able to get feedback from the project's stakeholders (primarily the client and the end users) and other parts of the project, such as the application design, the application implementation, alpha testing, and beta testing of the application, the feedback will be used continuously to improve this project and changes will be a part of the success of this project. Therefore, we will not describe every change between the different versions but only the major changes that have been made between the different versions as well as changes that have a significant impact on the application and the project.

8.1 Major changes between version 1 and 2

8.1.1 General Requirements

New requirements

In version two the following list of general requirements have been added to improve the usability of the application.

1. *2.3.22 Game variants*
2. *2.3.23 Scoreboard display*
3. *2.3.24 Skip game event*
4. *2.3.26 Game instructions*

Removed requirements

Some requirements have been moved to the design document, especially requirements that had to do with progressive web application. These requirements had a premature design and should not have been in version 1 of this document.

New color-palette

The client has provided a new color palette to be used in the application's GUI, see section *3.1.1 Look & Feel*.

Clarification of requirements

We have also clarified some information about the requirements according to the feedback. For example, the feedback asked how requirement risk evaluation is done, and from this we saw a need to explain not just this part but the other parts of the requirements as well, and therefore there is now two templates (one for functional and one for nonfunctional requirements) with explanations in the beginning of the section (*2.2.1 Template functional requirement* and *2.2.2 Template nonfunctional requirement*).

In addition to above we have added more detailed requirements to the sections Party, Trivia, Spin the Wheel, and Back to Back. For example four new separate requirements that describe how each game

module should pick a winner, according to the gameplay in the particular module, have been added, in addition to the General requirement *2.3.15 Game event winner(s)*.

8.1.2 Module Party

The Requirement *Choice of players* was renamed to *Number of players* and rephrased. A *Choice of winner* requirement was added as mentioned above.

8.1.3 Module Trivia requirements

Here the *Question Display* requirement was changed to *Game event display*, and the description rephrased to better reflect the intended functionality. We have also clarified, here and throughout the document, the words question, event, and many other terms that are used in the project.

8.1.4 Module Spin the Wheel requirements

The changes to this section consist of rephrasing the titles of the requirements and improving the descriptions of the requirements' functionality, in order to improve clarity concerning how the module is intended to work. However, the requirements are fundamentally the same since the previous version of this document.

8.1.5 Module Back to Back requirements

During the development process, we have developed four different game modules instead of the originally listed three. The fourth module is called Back to Back, and therefore a section with this module's requirements has been added.

8.2.1 Domain rules

4.2.4 Display final score and price was removed, as the way the application works at this point in time is that the game, once started, continues endlessly until the user either closes the application, reloads the page, or clicks a reset button that is present in the UI. However, score and prize display is still relevant in several of the requirements listed in *2. System-wide requirements*.

9 References

1. Marc Brysbaert, How many words do we read per minute? A review and meta-analysis of reading rate, chapter A meta-analysis of oral reading rates.

2. Jakob Nielsen, Usability Engineering, Chapter 5 The 3 Important Limits