Technical Specifications

Release 0.1.0

Alexander Faßbender, Alexander Preiß, Elias Ebi, Manuel Werner

Oct 06, 2020

TABLE OF CONTENTS:

CHAPTER

ONE

QUALITY QUEST

1.1 Introduction

In this project a serious game is to be developed, in which the audience can interactively determine the decisions in the course of the game. At the beginning of the game the audience can choose from several player characters. The subsequent decisions made by voting influence the further course of the game, either by improving skills and thus better chances for random decisions or by the consequences of bad decisions. The goal is to be invited to a prestigious workshop in Hawaii through good work.

1.2 Motivation

The image of the software developer is still strongly influenced by the cliché of the nerd living in his parents' basement who does whole projects on his own. However, due to the increasing focus on teamwork, readability and quality assurance, this has practically nothing in common with reality, so the game is intended to show the audience a contemporary picture of software development in a realistic and humorous way. The decisions made in the game are intended to demonstrate the skills required for this profession and show the positive effects of good practices in software development.

1.3 Vision

The game should have an appealing and clear presentation, and the spectators should be able to participate easily via their cell phones. The moderator should be able to continue the game if there are connection problems, and to pause voting if needed. The story should be played through within twenty minutes.

1.4 Context of project

QualityQuest is developed within a software project of the University of Ulm in cooperation with NewTec, represented by Dr. Axel Newe, and shall be published as OpenSource.

CHAPTER

TWO

GLOSSARY

The glossary contains a list of specific terms and their meaning in the context of the project. They can also contain examples, a description of what they are/can be.

2.1 Actors and roles

This section includes all actors involved in the system. Actors are people, but also third party technical systems involved in the system.

Term	Moderator
DESCRIPTION	Observe and comment on the game. If the online voting system fails, the moderator can take over the decisions and bring the game to an end.
IS-A	Human
CAN-BE	•
EXAMPLE	•

Term	PlayerAudience
DESCRIPTION	Viewers playing the game through StoryFlowDecisions.
IS-A	Human
CAN-BE	•
EXAMPLE	•

Term	Participants
DESCRIPTION	A person that participates and interacts with the game in any way.
IS-A	Human
CAN-BE	Moderator, PlayerAudience
EXAMPLE	•

Term	Customer
DESCRIPTION	The customer requires that the product meets certain requirements and is the first point of contact for questions and feedback.
IS-A	Human
CAN-BE	•
EXAMPLE	•

Term	Moderator-Client
DESCRIPTION	The Moderator-Client offers a graphical interface through which Moderator and PlayerAudience can interact with the game. The Moderator-Client establishes a connection to the server, which receives input from the PlayerAudience and visualises and logically implements the output of the server.
IS-A	Component
CAN-BE	•
EXAMPLE	•

Term	PlayerAudience-Client
DESCRIPTION	The PlayerAudience-Client provides a graphical interface through which PlayerAudience can interact with the server to participate in polls. The PlayerAudience-Client establishes a direct connection to the server via a web interface.
IS-A	Component
CAN-BE	•
EXAMPLE	•

Term	Client
DESCRIPTION	A client serves as a graphical interface through which a participant can interact with the server and the game.
IS-A	Component
CAN-BE	Moderator-Client, PlayerAudience-Client
EXAMPLE	•

Term	Server
DESCRIPTION	The server serves as an interface for the communication between the Moderator-Client and the PlayerAudience-Clients. The server contains the Voting-Tool.
IS-A	Component
CAN-BE	•
EXAMPLE	•

Term	Voting-Tool
DESCRIPTION	The voting tool is a logical unit, and part of the server, which decides which StoryBranch of the StroyFlow is chosen based on the input of the PlayerAudience-Clients.
IS-A	•
CAN-BE	•
EXAMPLE	•

2.2 Expertise

This section contains a collection of information regarding technical terms that are used in the context of the project.

Term	StoryFlowDecision
DESCRIPTION	A special event within the game where the PlayerAudience needs to take a decision which influences the further StoryFlow. After a StoryFlowDecision the PlayerCharacterStatusValues can increase by several levels. The PlayerAudience decides through OnlineVoting.
IS-A	•
CAN-BE	•
EXAMPLE	•

Term	StoryFlow
DESCRIPTION	The actual flow of the game-story. The Story contains elements where the PlayerAudience needs to make a StoryFlowDecision and depending on the decision, the StoryFlow progresses in different StoryBranches.
IS-A	•
CAN-BE	•
EXAMPLE	•

Term	StoryBranch
DESCRIPTION	A branch of the non-linear StoryFlow of the game.
IS-A	•
CAN-BE	•
EXAMPLE	•

Term	PlayerCharacterStatusValue
DESCRIPTION	The PlayerCharacter has different character status values which improve or change during the course of the game. The PlayerCharacterStatusValues are displayed via a PlayerCharacterStatusBox.
IS-A	•
CAN-BE	Programming, Analytics, Communication, Partying
EXAMPLE	•

Term	Role-playing game
DESCRIPTION	QualityQuest is a role-playing game. A role-playing game is a game in which players assume the roles of characters in a fictional setting.
IS-A	•
CAN-BE	QualityQuest
EXAMPLE	•

Term	PlayerCharacterStatusBox
DESCRIPTION	An info box that displays the different PlayerCharacterStatusValues and the portrait of the PlayerCharacter. The box can be displayed for example in the lower left corner.
IS-A	•
CAN-BE	•
EXAMPLE	•

Term	PlayerCharacter
DESCRIPTION	The virtual representation of the PlayerAudience in the game. At the beginning of the game the PlayerAudience chooses a PlayerCharacter from a collection of predefined PlayerCharacters with different PlayerCharacterStatusValues. The PlayerCharacter has different PlayerCharacterStatusValues and a portrait.
IS-A	•
CAN-BE	•
EXAMPLE	•

Term	Randomness
DESCRIPTION	The progression in the StoryFlow after a StoryFlowDecision is decided by on of the following randomness options: - ZeroRandomness - DiceRandomness
IS-A	•
CAN-BE	ZeroRandomness, DiceRandomness
EXAMPLE	•

Term	ZeroRandomness
DESCRIPTION	The StoryFlowDecision leads directly to the next StoryBranch. The random element is zero.
IS-A	Randomness
CAN-BE	•
EXAMPLE	•

Term	DiceRandomness
DESCRIPTION	After a StoryFlowDecision a die is rolled, which initiates the further StoryFlow and selects the next StoryBranch.
IS-A	Randomness
CAN-BE	•
EXAMPLE	•

Term	Programming
DESCRIPTION	A status value of the PlayerCharacter. Influences how well the character can program, for example less time is needed to program tests.
IS-A	CharacterStatusValue
CAN-BE	•
EXAMPLE	PlayerCharacter James has the programming-skill at 8.

Term	Analytics
DESCRIPTION	Determines how well the character can analyze situations and tasks, which increases the chance of success.
IS-A	CharacterStatusValue
CAN-BE	•
EXAMPLE	4 of 6 DiceRandomness possibilities lead to a positive event, because of high analytic stats.

Term	Communication
DESCRIPTION	Communication is a StatusValue of the PlayerCharacter. Communication influences how eloquent the PlayerCharacter is, e.g how well he works in a team or how well he deals with customers.
IS-A	CharacterStatusValue
CAN-BE	•
EXAMPLE	•

Term	Partying
DESCRIPTION	A character with a good partying skill can make more contacts at a party more quickly.
IS-A	CharacterStatusValue
CAN-BE	•
EXAMPLE	•

Term	OnlineVoting
DESCRIPTION	The PlayerAudience selects its decisions for a StoryFlowDecision via an online voting system. The Connection with the OnlineVoting is established by a QR-Code.
IS-A	•
CAN-BE	•
EXAMPLE	•

Term	Sidekick-Pet
DESCRIPTION	Can be unlocked by the PlayerAudience through a StoryFlowDecision. Helps the player in StoryFlowDecisions with helpful tips and suggestions.
IS-A	•
CAN-BE	•
EXAMPLE	•

Term	Play-Time
DESCRIPTION	The time it takes to finish a game. The time needed for QualityQuest should be about 15 to 20 minutes.
IS-A	
CAN-BE	•
EXAMPLE	•

Term	Voting-Timer
DESCRIPTION	Timer that is triggered by a StoryFlowDecision. While the timer is running the PlayerAudience has to vote. The moderator can stop the timer with the pause button.
IS-A	•
CAN-BE	•
EXAMPLE	The PlayerAudience has 60 seconds to vote on a StoryFlowDecision.

Term	Pause-Button
DESCRIPTION	Button with which the Voting-Timer can be stopped.
IS-A	•
CAN-BE	•
EXAMPLE	•

CHAPTER

THREE

REQUIREMENTS

The requirements are divided into different priorities, whose meaning should be clear from the following table:

PRIORITY	DESCRIPTION
+	The requirement must be fulfilled in any case so that the product can be accepted.
0	The fulfillment of the requirement is optional and therefore not necessarily a prerequisite for acceptance, but would have a very positive effect on the product.
-	The fulfilment of the requirement is also optional and therefore not a prerequisite for the acceptance.

3.1 Functional requirements

This section contains all requirements that specify the basic actions of the software system.

REQUIREMENT	Game type
ID	FA1
PRIORITY	+
DESCRIPTION	QualityQuest shall be a 2D RPG.
EXPLANATION	The PlayerAudience takes over the decision of a character in a fictional world of a software engineer. The PlayerAudience plays the game only through StoryFlowDecisions, for example the game plays like a movie in which the PlayerAudience takes over the decisions of the main character.

REQUIREMENT	Stand-alone game
ID	FA2
PRIORITY	+
DESCRIPTION	QualityQuest shall be a stand-alone game.
EXPLANATION	is means that the final binaries shall include everything that is needed to run the game. Any possibly needed framework needs to be included. The installation of additional frameworks or libraries is not acceptable.

REQUIREMENT	Game presentation
ID	FA3
PRIORITY	+
DESCRIPTION	QualityQuest shall be a visual-based 2D RPG.
EXPLANATION	This means that QualityQuest shall not be a purely text-based game, but text may be an element of the visual appearance of the game.

REQUIREMENT	NewTec branding
ID	FA4
PRIORITY	+
DESCRIPTION	QualityQuest shall display the NewTec logo clearly visible all the time.
EXPLANATION	•

REQUIREMENT	Game language
ID	FA5
PRIORITY	+
DESCRIPTION	The main language of QualityQuest shall be German.
EXPLANATION	The majority of in-game language shall be German, but typical software engineering terms that are not German, but are commonly used in Germany do not need to be translated.

REQUIREMENT	Game language options
ID	FA6
PRIORITY	-
DESCRIPTION	QualityQuest should support multiple languages.
EXPLANATION	•

REQUIREMENT	Music
ID	FA7
PRIORITY	-
DESCRIPTION	QualityQuest may be accompanied by a suitable musical background to enhance the player experience.
EXPLANATION	•

REQUIREMENT	Sound effects
ID	FA8
PRIORITY	0
DESCRIPTION	QualityQuest should emphasize important events of the StoryFlow with sound effects.
EXPLANATION	•

REQUIREMENT	Game content
ID	FA9
PRIORITY	+
DESCRIPTION	QualityQuest shall tell a story which mainly consists of typical elements of the software engineering profession.
EXPLANATION	•

REQUIREMENT	StoryFlow
ID	FA10
PRIORITY	+
DESCRIPTION	The story of QualityQuest shall be non-linear.
EXPLANATION	The story shall contain elements where the PlayerAudience needs to make a StoryFlowDecision. Depending on the decision, the StoryFlow shall continue in different StoryBranches.

REQUIREMENT	Influence on the StoryFlow by the player
ID	FA11
PRIORITY	+
DESCRIPTION	The PlayerAudience shall influence the selection of StoryBranches by means of StoryFlowDecisions.
EXPLANATION	•

REQUIREMENT	Participation of a larger PlayerAudience
ID	FA12
PRIORITY	+
DESCRIPTION	QualityQuest shall enable the PlayerAudience to let a larger audience participate in StoryFlowDecisions by means of OnlineVoting.
EXPLANATION	It would be highly desirable that the OnlineVoting feature is directly embedded into the game. Other methods are acceptable depending on the circumstances.

REQUIREMENT	Random element of StoryFlow control
ID	FA13
PRIORITY	+
DESCRIPTION	The selection of a StoryBranch after a StoryFlowDecision shall be generated randomly.
EXPLANATION	Randomness can be either determined through ZeroRandomness or DiceRandomness.

REQUIREMENT	Visualizing the randomness
ID	FA14
PRIORITY	+
DESCRIPTION	If the selection of a StoryBranch after a StoryFlowDecision is generated with DiceRandomness, QualityQuest shall display a clear visualization of the randomization-process.
EXPLANATION	•

REQUIREMENT	Character status values
ID	FA15
PRIORITY	+
DESCRIPTION	The PlayerCharacter shall have different status values, which can improve or worsen during the game. The PlayerCharacter shall have all of the following status values: Programming, Analytics, Communication, Partying.
EXPLANATION	•

REQUIREMENT	Selecting a character
ID	FA16
PRIORITY	+
DESCRIPTION	At the start of the game the PlayerAudience shall choose a PlayerCharacter from a selection of possible PlayerCharacters via the voting system.
EXPLANATION	•

REQUIREMENT	Presentation of character status values
ID	FA17
PRIORITY	+
DESCRIPTION	The first StoryFlowDecision shall be the selection of the PlayerCharacter.
EXPLANATION	This shall be a StoryFlowDecision with ZeroRandomness.

REQUIREMENT	Portrait of the PlayerCharacter
ID	FA18
PRIORITY	+
DESCRIPTION	QualityQuest shall display a portrait of the PlayerCharacter as part of the PlayerCharacterStatusBox all the time.
EXPLANATION	•

REQUIREMENT	Character levelling
ID	FA19
PRIORITY	+
DESCRIPTION	The PayerCharacter shall level up its status values based on events or StoryFlowDecisions.
EXPLANATION	•

REQUIREMENT	Visual presentation of PlayerCharacter status changes
ID	FA20
PRIORITY	+
DESCRIPTION	The change of status values of the PlayerCharacter shall be highlighted visually.
EXPLANATION	•

REQUIREMENT	Acoustic presentation of PlayerCharacter status changes
ID	FA21
PRIORITY	0
DESCRIPTION	The change of status values of the PlayerCharacter should be highlighted acoustically.
EXPLANATION	•

REQUIREMENT	Programming language
ID	FA22
PRIORITY	+
DESCRIPTION	QualityQuest shall be programmed in a C dialect (C, C++ or C#).
EXPLANATION	•

REQUIREMENT	Development environment
ID	FA23
PRIORITY	+
DESCRIPTION	Both the source code and the build solution of QualityQuest shall be buildable in one of the following development environments: Microsoft Visual Studio, Microsoft Visual Studio Code.
EXPLANATION	•

REQUIREMENT	Operating system
ID	FA24
PRIORITY	+
DESCRIPTION	QualityQuest shall run on Microsoft Windows 10 operating system.
EXPLANATION	•

REQUIREMENT	Usage of game engines
ID	FA25
PRIORITY	-
DESCRIPTION	An existing game engine may be used, if all of the following conditions apply: The license conditions of the game engine allow the source code of QualityQuest to be open source. The license conditions of the game engine allow the usage of the game engine without license fees. The license conditions of the game engine allow the usage of QualityQuest as intended by NewTec without license fees.
EXPLANATION	•

REQUIREMENT	Usage of online voting solutions
ID	FA26
PRIORITY	-
DESCRIPTION	An existing online voting solution may be used, if the license conditions of the online voting solution allow the usage of the online voting solution in the context of QualityQuest as intended by NewTec without license fees.
EXPLANATION	•

REQUIREMENT	Pause Game
ID	FA27
PRIORITY	+
DESCRIPTION	The moderator shall have the possibility to pause the game with the PauseButton. The PauseButton shall be around the lower right edge.
EXPLANATION	•

REQUIREMENT	Connection Setup
ID	FA28
PRIORITY	+
DESCRIPTION	The server shall allow as many PlayerAudience-Clients as possible to connect to the game via the network. However, the server should only allow a single Moderator-Client to connect to the server at any given time. Once the Moderator-Client established the connection to the server, the Moderator has the option to start or interrupt the game at any time.
EXPLANATION	•

REQUIREMENT	Connection Timeout
ID	FA29
PRIORITY	+
DESCRIPTION	If the Moderator-Client does not send back an ACK within a certain time-frame after recieving the server's message, the connection to the server shall be interrupted. In this case the Moderator can either continue playing in offline mode or try to re-establish the connection to the server.
EXPLANATION	This serves as a fail-save, for the case that messages/ACKs could be corrupted or the connection to the server is lost.

REQUIREMENT	Server connection loss
ID	FA30
PRIORITY	+
DESCRIPTION	If a Moderator-Client or PlayerAudience-Client loses its connection to the server, its Unique User Identifier (UUID) shall be stored in the system. In this case, the respective client can reconnect to the server to participate in the game again.
EXPLANATION	•

REQUIREMENT	Data exchange file format
ID	FA31
PRIORITY	+
DESCRIPTION	The file format for data exchange between clients and server shall be JSON.
EXPLANATION	•

REQUIREMENT	Unique User Identifier (UUID)
ID	FA32
PRIORITY	+
DESCRIPTION	Every participant shall be assigned an Unique User Identifier (UUID) based on either their IP-address or MAC-address.
EXPLANATION	This ensures participants can rejoin the game after leaving the game or losing the connection to the server.

REQUIREMENT	Offline-Mode
ID	FA33
PRIORITY	+
DESCRIPTION	In the event that the server is not functional, the network infrastructure slows significantly down or there being a problem with the connection between clients and server, the Moderator shall have the option to continue the game offline. This Offline-Mode must ensure a smooth transition between online and offline and shall be able to step in at any time.
EXPLANATION	•

REQUIREMENT	Game Engine
ID	FA34
PRIORITY	+
DESCRIPTION	As a game engine the project shall use Unity.
EXPLANATION	•

REQUIREMENT	Communication protocol
ID	FA35
PRIORITY	+
DESCRIPTION	The communication protocol shall define clearly and well-defined how clients and server shall communicate with each other in order to accept messages. If a client increasingly does not adhere to the communication protocol, a communication protocol violation is detected.
EXPLANATION	This ensures that it is not easily possible to tinker with the game through an altered client.

REQUIREMENT	Communication protocol violation
ID	FA36
PRIORITY	+
DESCRIPTION	If a client increasingly does not adhere to the communication protocol, the UUID of the participant should be excluded from the rest of the game.
EXPLANATION	This ensures that it is not easily possible to tinker with the game through an altered client.

3.2 Non-functional Requirements

This section specifies the non-functional requirements for the software system.

REQUIREMENT	Documents to be delivered
ID	QA1
PRIORITY	+
DESCRIPTION	A System Specification, which comprises use case diagrams, use case descriptions and a static view of the software architecture and Software Design Specification for each software component, which describes both the static and the dynamic view shall be delivered.
EXPLANATION	•

REQUIREMENT	In-code documentation style
ID	QA1
PRIORITY	+
DESCRIPTION	The source code shall be documented by means of Doxygen and in Javadoc style.
EXPLANATION	•

REQUIREMENT	In-code documentation content
ID	QA3
PRIORITY	+
DESCRIPTION	All of the following source code elements shall be documented: Constants, variables and defines. Classes and class members. Methods and method signatures, including return values. Functions and function signatures, including return values.
EXPLANATION	•

REQUIREMENT	Documentation style for diagrams
ID	QA4
PRIORITY	+
DESCRIPTION	All documentation diagrams shall follow the UML standard.
EXPLANATION	•

REQUIREMENT	Delivery of UML diagrams
ID	QA5
PRIORITY	+
DESCRIPTION	All UML diagrams shall be delivered in the form of a diagram and a PlantUML link.
EXPLANATION	•

REQUIREMENT	Adherence to project Coding Styleguide
ID	QA6
PRIORITY	0
DESCRIPTION	The software code should adhere to the Project Coding Styleguide.
EXPLANATION	•

REQUIREMENT	Adherence to Clean Code Principles
ID	QA7
PRIORITY	0
DESCRIPTION	The software code should adhere to Grade 1 (Red) of the Clean Code Principles.
EXPLANATION	•

REQUIREMENT	Target PlayerAudience
ID	QA8
PRIORITY	+
DESCRIPTION	QualityQuest shall address a target audience of university students with interest in a SW engineering career.
EXPLANATION	•

REQUIREMENT	Playing time
ID	QA9
PRIORITY	+
DESCRIPTION	The complete story of QualityQuest shall be playable in a time frame of 15 to 20 minutes.
EXPLANATION	•

REQUIREMENT	Playing fun
ID	QA10
PRIORITY	0
DESCRIPTION	The story of QualityQuest should be humorous.
EXPLANATION	•

REQUIREMENT	Player motivation
ID	QA11
PRIORITY	+
DESCRIPTION	The audience of QualityQuest shall be encouraged to follow the story by motivational elements.
EXPLANATION	Motivational elements could be for example rewards, achievement & level upgrades.

REQUIREMENT	Deliverable artefacts
ID	QA12
PRIORITY	+
DESCRIPTION	Documentation, Source Code and a running version of QualityQuest shall be delivered to NewTec.
EXPLANATION	•

REQUIREMENT	Type of delivery
ID	QA13
PRIORITY	+
DESCRIPTION	All deliverable artifacts shall be delivered digitally.
EXPLANATION	The delivery can be by depositing the deliverable artefacts in a public version control system. Documents should be delivered in both PDS and DOCX.

REQUIREMENT	Deadline
ID	QA14
PRIORITY	+
DESCRIPTION	The deadline for the final delivery is 2021-04-28.
EXPLANATION	•

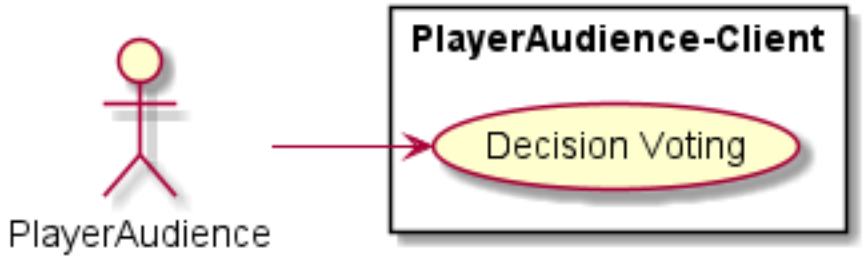
REQUIREMENT	Open source development
ID	QA15
PRIORITY	-
DESCRIPTION	The Source Code of QualityQuest may be published open source under CreativeCommons CC BY-NC 4.0 license terms.
EXPLANATION	•

CHAPTER

FOUR

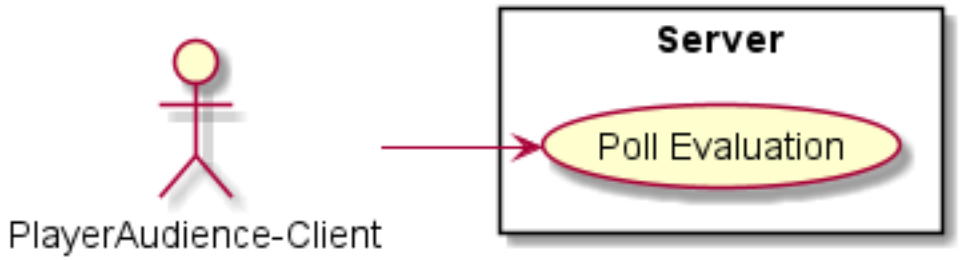
USE-CASE DIAGRAMS

4.1 PlayerAudience-Client



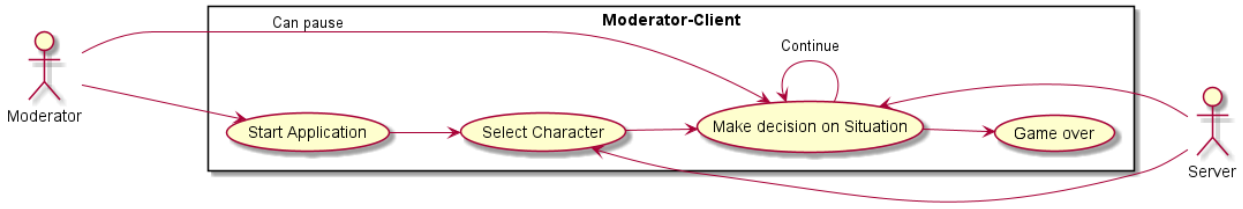
Via the PlayerAudience-Client, the individual participant, as part of the audience, has the possibility to vote on the decisions that occur in the game. The PlayerAudience-Client also acts as an actor for the server.

4.2 Server



Server receives the respective decisions from the PlayerAudience-Client, collects and evaluates them in relation to the number of votes. The server also acts as an actor for the Moderator-Client.

4.3 Moderator-Client



The Server tells the Moderator-Client what decisions the audience has made. The Moderator is responsible for starting the game and can start and pause the voting process at any given time. A decision is always followed by a new decision until the game eventually ends.

CHAPTER

FIVE

ARCHITECTURE DIAGRAMS

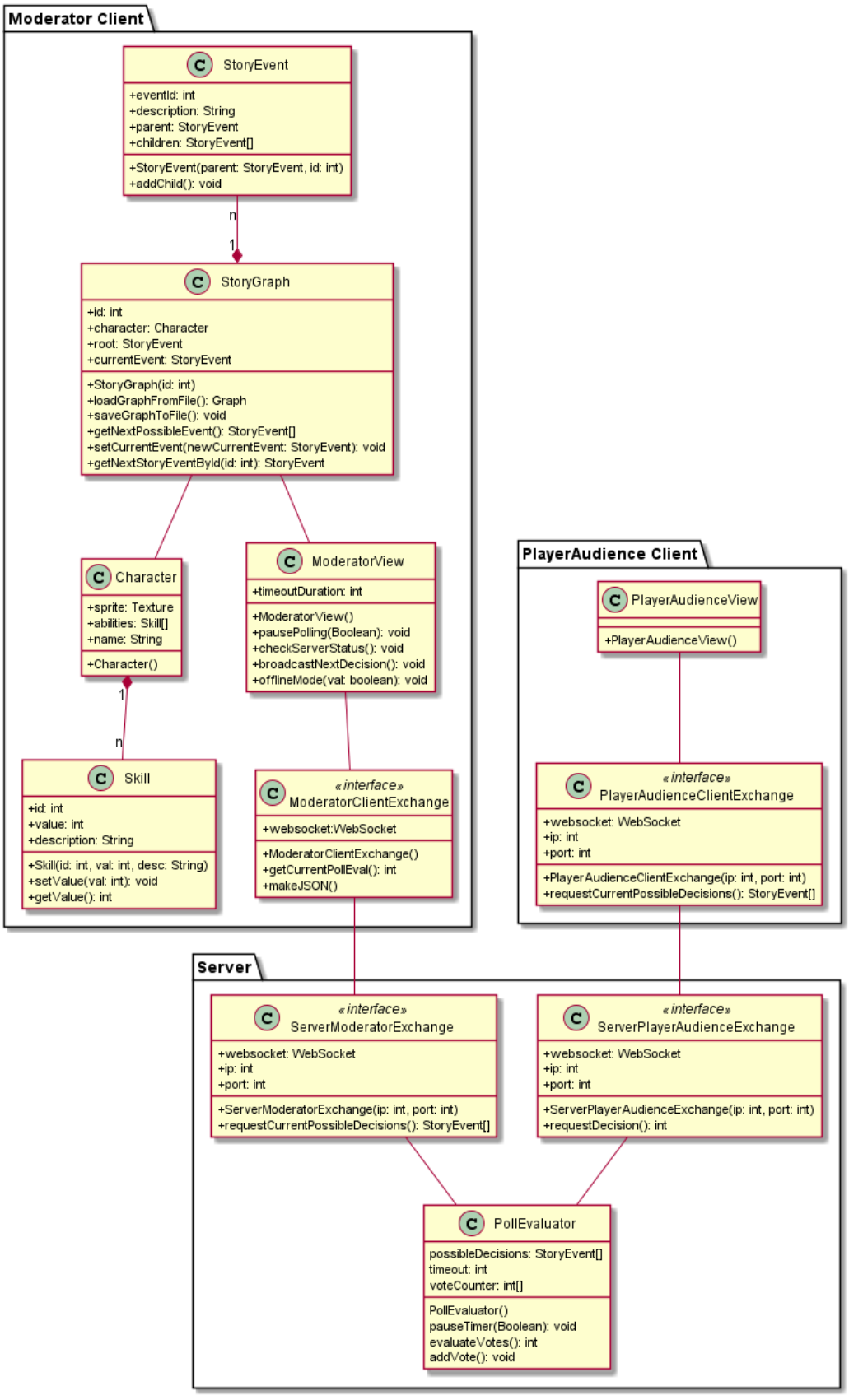
The diagrams below show the architectural structure of the different components of QualityQuest and how they communicate via well-defined interfaces.

5.1 Class diagram

The class diagram shows the architectural structure of the individual components of QualityQuest and thus which classes are used in the implementation to logically implement the project. It also shows which interfaces the individual components use to communicate with each other.

Since the StoryGraph is run and build locally on the Moderator-Client and the server only has a forwarding and poll evaluation function (possible decisions get forwarded to the audience, polls get evaluated, result of the vote gets forwarded to the Moderator-Client), a fallback is easily possible with only the Moderator as decisionmaker. So if the server is no longer accessible for the Moderator-Client, this will be noticed by the regular status requests and the Moderator will be informed together with the option to switch to Offline-Mode. Since the server only informs the Moderator-Client about the results of the voting, in case of a server failure, only the information about the voting conditions is lost for the Moderator-Client, because the actual decisions are made locally.

In Offline-Mode, the timer is deactivated and the Moderator can select decisions directly, while status checks are still performed in the background to inform the Moderator in case the server becomes available again.

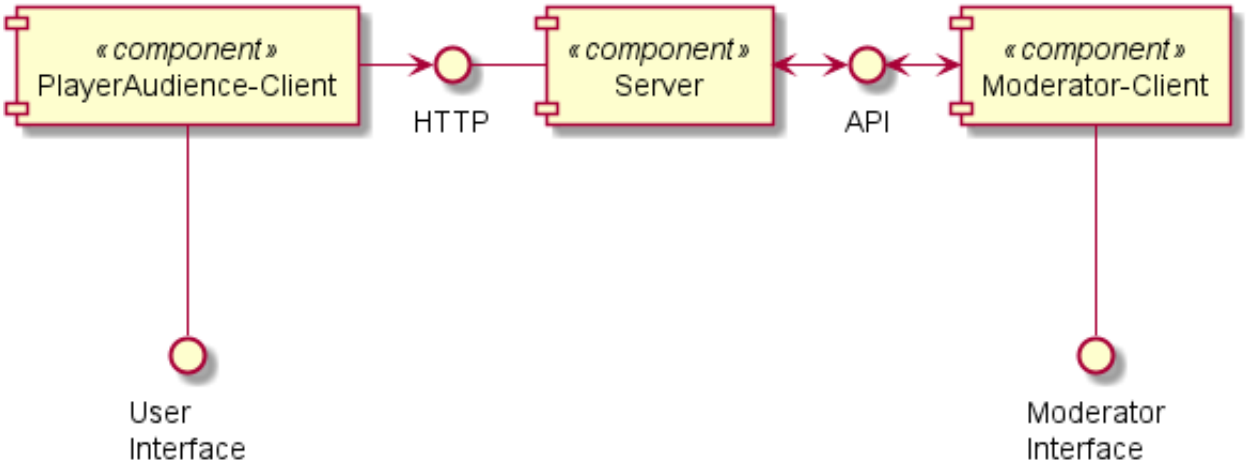

Moderator Client
StoryEvent
+eventId: int
+description: String
+parent: StoryEvent
+children: StoryEvent[]
+StoryEvent(parent: StoryEvent, id: int)
+addChild(): void
n
1
StoryGraph
+id: int
+character: Character
+root: StoryEvent
+currentEvent: StoryEvent
+StoryGraph(id: int)
+loadGraphFromFile(): Graph
+saveGraphToFile(): void
+getNextPossibleEvent(): StoryEvent[]
+setCurrentEvent(newCurrentEvent: StoryEvent): void
+getNextStoryEventById(id: int): StoryEvent
Character
+sprite: Texture
+abilities: Skill[]
+name: String
+Character()
1
n
Skill
+id: int
+value: int
+description: String
+Skill(id: int, val: int, desc: String)
+setValue(val: int): void
+getValue(): int
ModeratorView
+timeoutDuration: int
+ModeratorView()
+pausePolling(Boolean): void
+checkServerStatus(): void
+broadcastNextDecision(): void
+offlineMode(val: boolean): void
«interface»
ModeratorClientExchange
+websocket:WebSocket
+ModeratorClientExchange()
+getCurrentPollEval(): int
+makeJSON()
PlayerAudience Client
PlayerAudienceView
+PlayerAudienceView()
«interface»
PlayerAudienceClientExchange
+websocket: WebSocket
+ip: int
+port: int
+PlayerAudienceClientExchange(ip: int, port: int)
+requestCurrentPossibleDecisions(): StoryEvent[]
Server
«interface»
ServerModeratorExchange
+websocket: WebSocket
+ip: int
+port: int
+ServerModeratorExchange(ip: int, port: int)
+requestCurrentPossibleDecisions(): StoryEvent[]
«interface»
ServerPlayerAudienceExchange
+websocket: WebSocket
+ip: int
+port: int
+ServerPlayerAudienceExchange(ip: int, port: int)
+requestDecision(): int
PollEvaluator
possibleDecisions: StoryEvent[]
timeout: int
voteCounter: int[]
PollEvaluator()
pauseTimer(Boolean): void
evaluateVotes(): int
addVote(): void

5.2 Component diagrams

The class diagrams show the architectural structure of the individual components using interfaces/ports and subsystems.

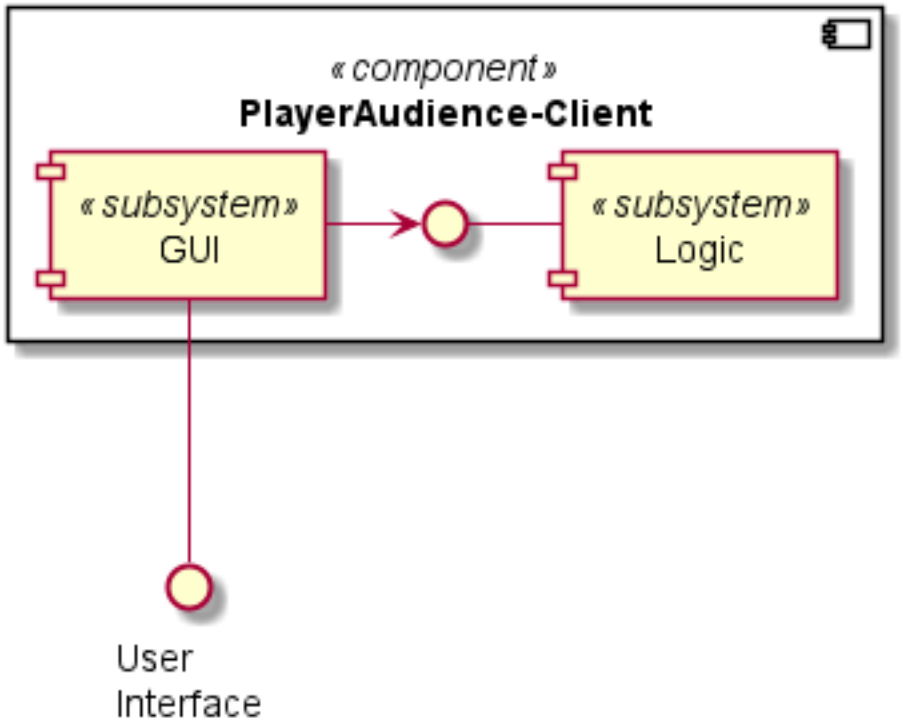
5.2.1 Component-Overview

An overview of all components of QualityQuest and which interfaces exist between the individual components, or the user interfaces of the participants.



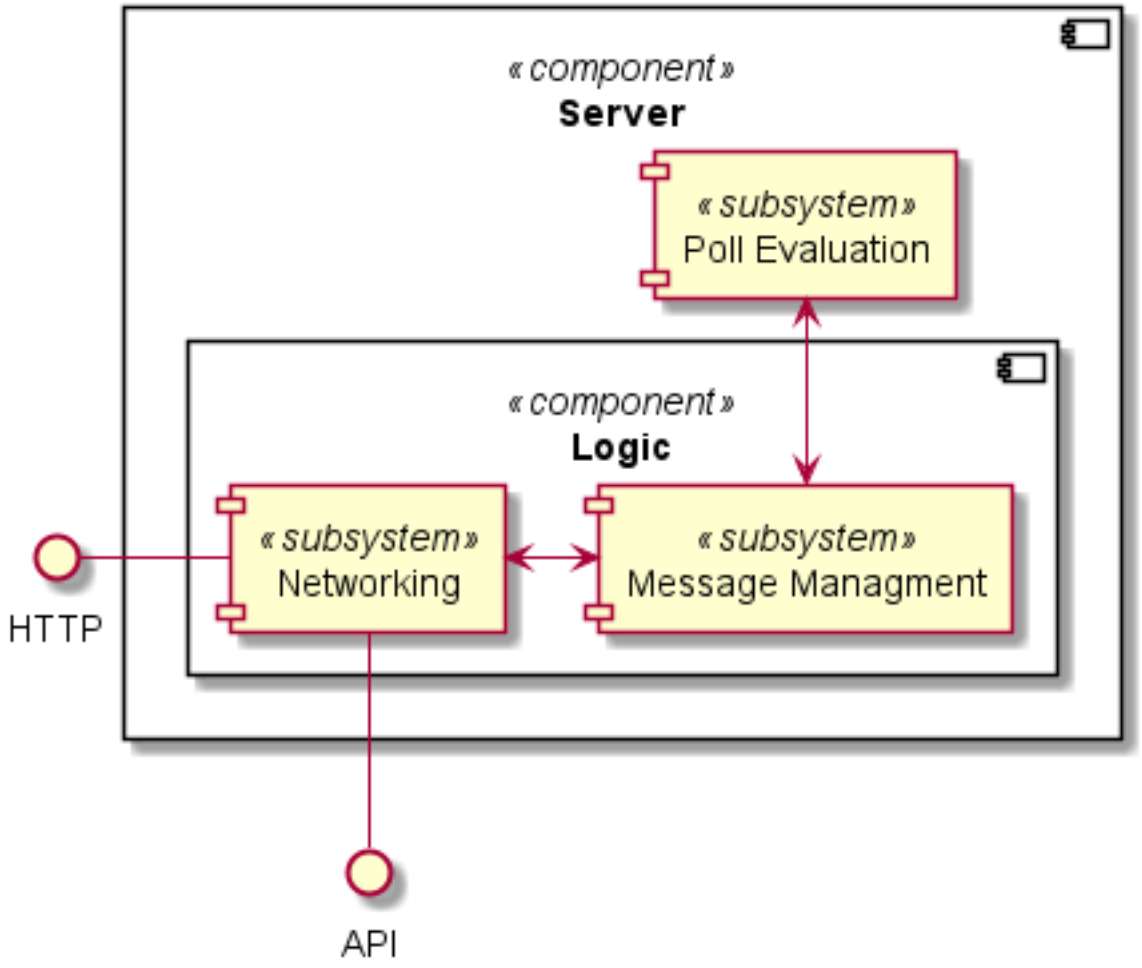
5.2.2 PlayerAudience-Client

Architectural overview of which subsystems and interfaces the PlayerAudience-Client component consists of.



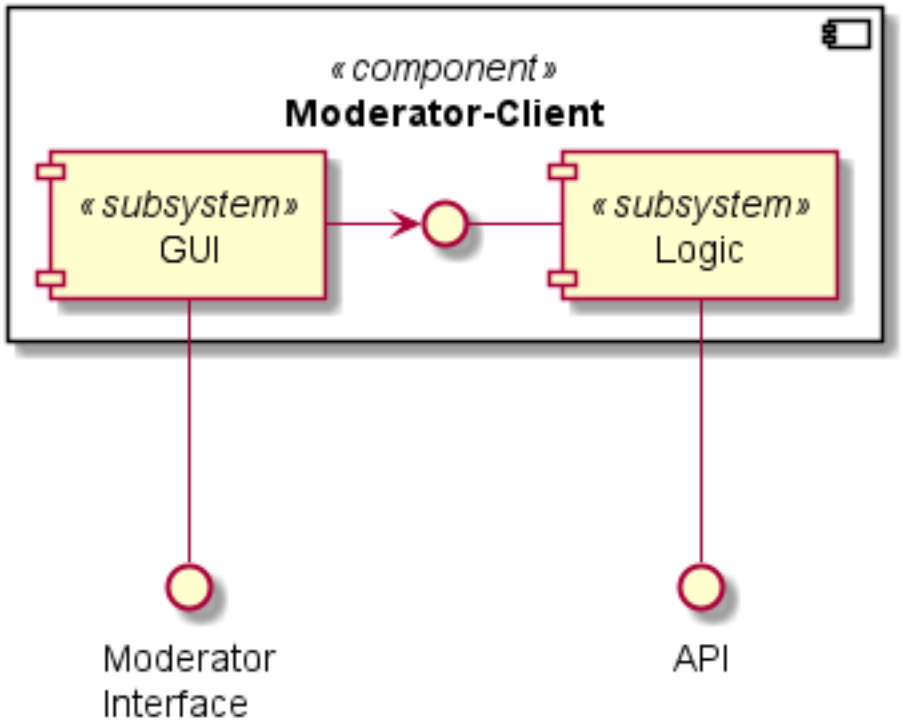
5.2.3 Server

Architectural overview of which subsystems and interfaces the Server component consists of.



5.2.4 Moderator-Client

Architectural overview of which subsystems and interfaces the Moderator-Client component consists of.

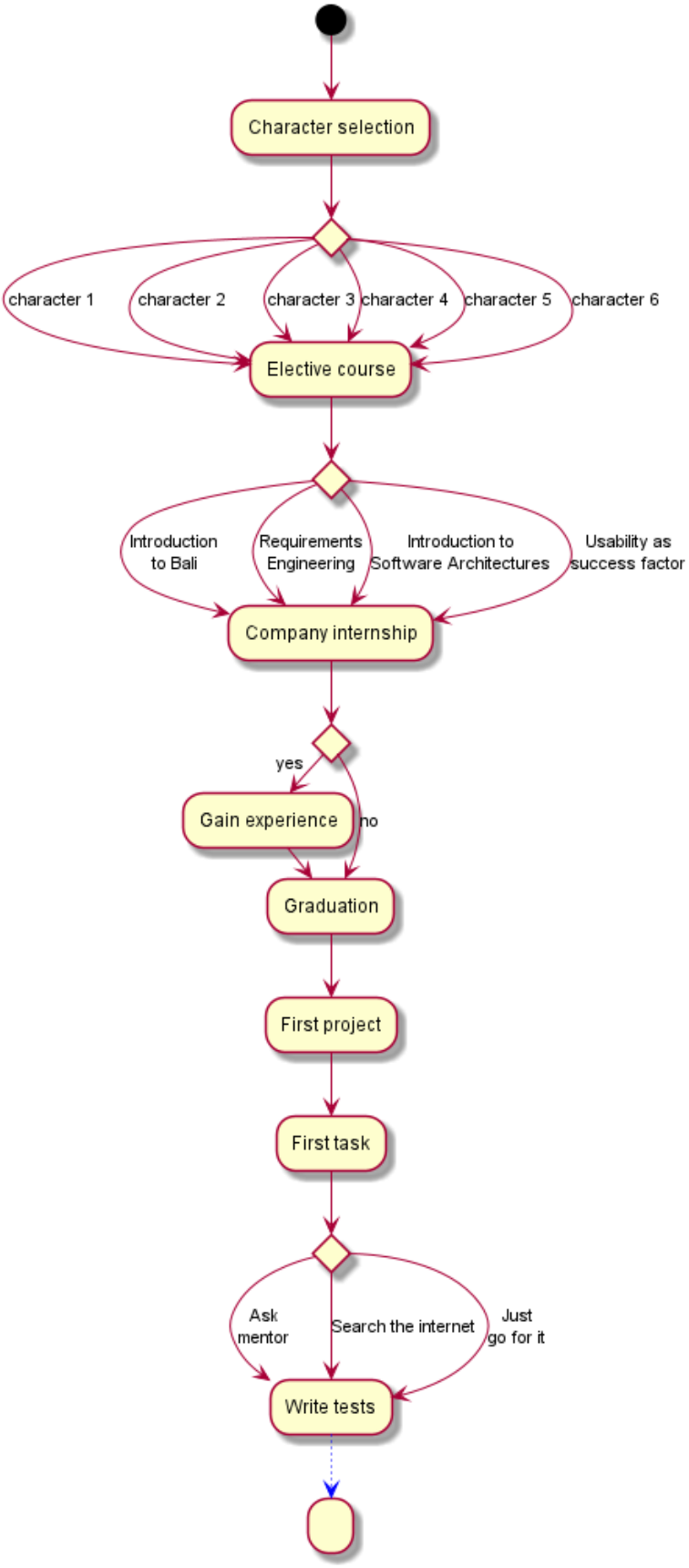

«component»
Moderator-Client
«subsystem»
GUI
«subsystem»
Logic
Moderator
Interface
API

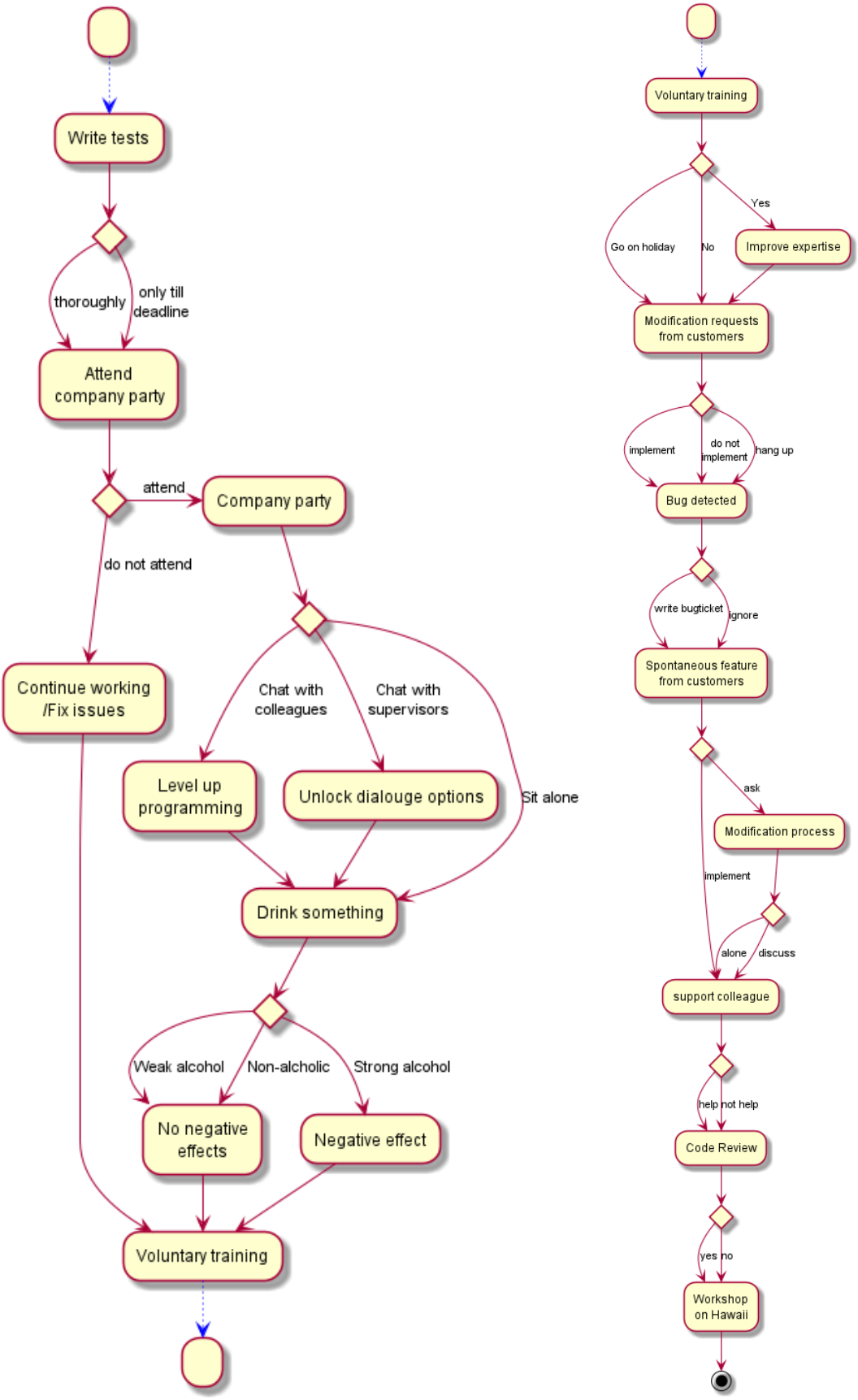
CHAPTER

SIX

STORYFLOW DIAGRAM

A rough overview of the decisions occurring within the story. The diagram exclusively depicts the situations in which decisions can be made and the choices to be made in these situations.


Character selection
character 1
character 2
character 3
character 4
character 5
character 6
Elective course
Introduction to Bali
Requirements Engineering
Introduction to Software Architectures
Usability as success factor
Company internship
yes
Gain experience
no
Graduation
First project
First task
Ask mentor
Search the internet
Just go for it
Write tests


Write tests
thoroughly
only till deadline
Attend company party
attend
Company party
do not attend
Continue working /Fix issues
Chat with colleagues
Chat with supervisors
Level up programming
Unlock dialouge options
Sit alone
Drink something
Weak alcohol
Non-alcholic
Strong alcohol
No negative effects
Negative effect
Voluntary training
Voluntary training
Yes
Go on holiday
No
Improve expertise
Modification requests from customers
implement
do not implement
hang up
Bug detected
write bugticket
ignore
Spontaneous feature from customers
ask
Modification process
implement
alone
discuss
support colleague
help not help
Code Review
yes no
Workshop on Hawaii

CHAPTER

SEVEN

USED TOOLS, PLUGINS AND LIBRARIES

A list of all tools, plugins and libraries that have been used to create the documentation of the product and helped implementing Quality Quest. The list contains the name of the respective tool, the version that has been used (or a link to the webpage) and a description on how the tool was used.

7.1 Used tools

The following tools were used during the creation of Quality Quest and the documentation:

Tool	Description	Version
Visual Studio Code	Tool to create and edit markdown files, and also to create and edit PlantUML files.	1.49
PlantUML	Used to create the required UML diagrams..	1.2020.15
GIMP	Editor for raster graphics. Was used for image editing.	2.10.18
SourceTree	GUI for git.	4.0.1
GitKraken	GUI for git.	5.0.4
MkDocs	Documentation tool for creating stylised documentation through markdown files.	1.1.2
GitHub	Tool for source code management and version control.	link
Doxygen	Tool to generate software reference documentation.	1.8.20
Aseprite	Tool to create pixelart sprites and animations.	1.2.25
Planttext	A website for converting PlantUML files into svg files.	link
Docker	Tool used to use other tools directly without the need of installing anything,	19.03.13
m2r	Markdown to reStructuredText converter used for making the docs usable for Sphinx.	link
Sphinx	Documentation tool for creating stylised documentation through reStructured-Text.	3.2.1

7.2 Used plugins

The following plugins were used during the creation of Quality Quest and the documentation:

Plugin	Description	Version
Markdown All in One	Virtual Studio Code plugin used for creating MkDocs easier.	3.3.0
PlantUML	Virtual Studio Code plugin used for creating PlantUML diagrams directly in VSC and to see the diagram live-update.	2.13.13
mkdocs-with-pdf	MkDocs plugin used for creating a PDF out of the MkDocs documents.	0.7.5
Admonition	MkDocs extension used to create admonitions in the documentation.	link

7.3 Used libraries

The following libraries were used during the creation of Quality Quest:

Library	Description	Version

CHAPTER

EIGHT

CHANGE HISTORY

The change history is a chronologically ordered list of all changes between different documentation versions. The different versions are listed together with the release date and a link to the changelog of the version.

8.1 Quick reference

Version	Quick Description	Date	Link
0.1	Architecture Design	2020-10-09	*Link*

8.2 Version 0.1 - Architecture design

This is the initial version of the Technical Specifications and thus has no changelog. The next version, which will focus on the component/detailed design, will be the first version with a changelog.