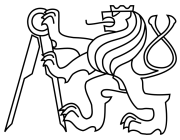
CZECH TECHNICAL UNIVERSITY IN PRAGUE
FACULTY OF INFORMATION TECHNOLOGY



# ASSIGNMENT OF MASTER'S THESIS

| | |
|---|---|
| **Title:** | Computer-aided system for cognitive training |
| **Student:** | Bc. Martin Paňko |
| **Supervisor:** | Mgr. Iveta Fajnerová, Ph.D. |
| **Study Programme:** | Informatics |
| **Study Branch:** | Web and Software Engineering |
| **Department:** | Department of Software Engineering |
| **Validity:** | Until the end of summer semester 2017/18 |

## Instructions

1. Design and implement a training system for cognitive functions using computer games in a form of desktop application according to following general requirements:
 - The system provides predefined modules for training of specific cognitive functions (e.g. attention, memory, and social abilities), each of them has a form of mini-game(s).
 - The system will manage individual patients' accounts and the training process (e.g., duration, frequency, difficulty level).
 - The system will maintain the database that stores data of individual games for each patient; enables CSV export for additional analysis.
 - The system provides basic visualization of the data obtained in individual games/modules.
2. Design and implement several mini-games in order to test usability and utility of the system. Specifications of mini-games to be implemented, will be given by the supervisor based on clinical requirements.
3. Demonstrate functionality of the system by adequate usability testing.

## References

Will be provided by the supervisor.

Ing. Michal Valenta, Ph.D.
Head of Department

prof. Ing. Pavel Tvrdík, CSc.
Dean

Prague February 8, 2017

# Acknowledgement / Declaration

I would like to thank my supervisor Mgr. et Mgr. Iveta Fajnerová, Ph.D. for her cooperation and advices. I would like to thank RNDr. Petr Olšák for his work on CTUstyle.

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended. In accordance with Article 46(6) of the Act, I hereby grant a nonexclusive authorization (license) to utilize this thesis, including any and all computer programs incorporated therein or attached thereto and all corresponding documentation (hereinafter collectively referred to as the "Work"), to any and all persons that wish to utilize the Work. Such persons are entitled to use the Work in any way (including for-profit purposes) that does not detract from its value. This authorization is not limited in terms of time, location and quantity.

In Prague on 7th of May 2018

.....

# Abstrakt / Abstract

Táto diplomová práca sa zaoberá dizajnom a implementáciou aplikácie, ktorá umožňuje jej používateľom vykonávať tréning v rôznych oblastiach kognície prostredníctvom hrania 3d minihier. Aplikácia spravuje používateľské účty rôznych rolí a poskytuje základnú funkcionalitu pre tréningový proces. Aplikácia ukladá herné výsledky pre každého používateľa z dôvodu ďalšej analýzy.

**Kľúčové slová:** Unity, Unity3d, C#, kognitívny tréning, 3d games

The aim of this thesis is to design and implement desktop application that enables the users to perform training in various areas of cognition via playing 3d minigames. The application manages accounts of various user roles and provides basic functionality for the training process. The application stores the game results for each user by reason of additional analysis.

**Keywords:** Unity, Unity3d, C#, cognitive training, 3d games

# Contents /

# Tables / Figures

# Chapter 1
# Introduction

## 1.1 Motivation

Neurological and neuropsychiatric disorders have devastating negative impact for individuals and communities worldwide. The patients having cognitive deficit or cognitive impairment as cause of schizophrenia, Alzheimer's disease, stroke, depression, etc. require complex medical service.

This thesis describes the process of developing functional prototype of computer application with the view of its future use as a part of complex clinical training for people suffering cognitive deficit. The application aims to support the training in clinical environment by allowing the patients to go through the training process using personal computer at their home. Making use of cognitive training via computer application aims to support the patients after ending their medical training in clinical facilities or when visiting such a facility is difficult eg. due to long distance to clients place of residence.

The application should guide and support the patients that are in the process of clinical treatment by cognitive training that is adjusted by their doctor or supervisor from clinical environment. The main medium for the application's cognitive training are minigames designed by neurologists.

The application is being developed under supervision and cooperation with Mgr. et Mgr. Iveta Fajnerová, Ph.D. of RP3 Applied Neurosciences and Brain Imaging in National Institute of Mental Health [1].

## 1.2 The Project

The aim of the project is to design and implement desktop application that will enable users to enhance their cognitive functions. The application will encompass four 3d minigames that are designed in cooperation with the thesis supervisor. The minigames are ordered into categories by the specific category of cognition.

The application should be designed in such a way, that it serves as a suitable framework for the following development that will extend the application by adding chat functionality, additional 3d minigames playable using Virtual Reality headsets and support for mobile and tablet platforms.

The application should allow its users to log in and offer them daily training routine visualizing their progress and supervision of their therapists.

In addition the application should allow researchers to obtain access to game results of application's users for for analysis and export to csv files. In addition to focusing on ease of use, the application should have attractive and fresh design in order to ensure smooth user experience even for less proficient users.

[1] official website of the institute is http://www.nudz.cz/en/

The process of the application development starts with analysis that provides insight by reviewing similar existing applications and continues with design that constructs fitting software architecture. The implementation phase describes the process of employing design and creating functional application that is inspected and evaluated in test phase.

# Chapter 2
# Analysis

This chapter describes the process of analysing the resources and information. At first, it provides the review of existing applications, continuing by description of personas. The subsequent sections describe functional, non-functional requirements for the application as well as use cases and scenarios for particular user roles.

## 2.1 Existing applications

An important step in process of developing new application is to review existing applications that solve similar problems. This often provides valuable information to learn from in pursuance to suitably formulate demands and requirements on developed application. In the following sections, multiple applications are reviewed and analysed, arranging information and describing their features, advantages and disadvantages.

### 2.1.1 Neurokog

| | |
|---|---|
| **Author:** | National Institute of Mental Health |
| **Platforms:** | Web Browser |
| **Updated:** | March 2018 |
| **Price:** | The application is not available to general public |
| **Website:** | https://neurokog.nudz.cz/ |

**Table 2.1.** Neurokog details

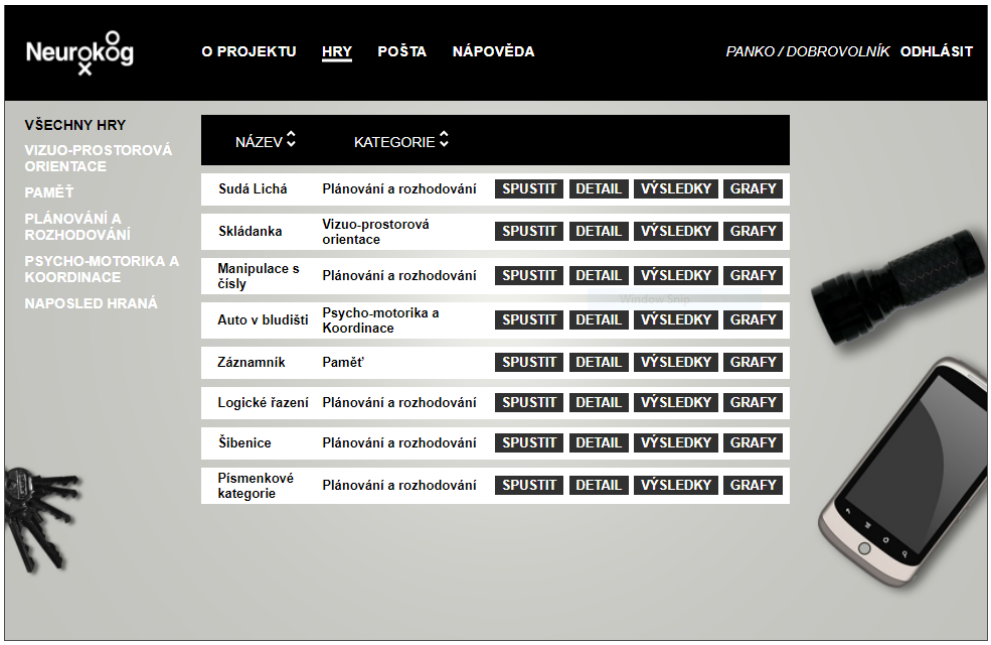



**Figure 2.1.** Screenshot of Neurokog

Neurokog is web application that is being developed in Workgroup on neurocognition led by PhDr. Mabel Virginia Rodriguez Manchola, Ph.D. in National Institute of Mental Health. The application is used as a tool in remediation for cognitively impaired individuals with diagnoses of neuropsychiatric circuit disorders. Neurokog serves the similar purpose as the application that is being developed by the author of this thesis but there are several limitations for Neurokog. The main limitation is, that Neurokog is a browser based application developed in javascript and therefore it does not enable to use system resources for accelerated 3d graphics. Therefore, Neurokog does allow to implement 3d minigames that are necessary for certain cognitive trainings, eg. training of Visuo-spatial Orientation. Furthermore, the absence of 3d minigames does not allow to employ the concept of Virtual Reality for cognitive training. The main disadvantages of Neurokog are as follows:

- Does not enable to use 3d games and VR,
- has unattractive UI design, using conservative web elements,
- window layout does not scale properly on small displays (eg. mobile browser)
- poor visualisation of game results,
- lacks any motivational elements for daily training

### 2.1.2 Lumosity

| | |
|---|---|
| **Author:** | Lumos Labs, Inc. |
| **Platforms:** | Android, iOS, Web Browser |
| **Price:** | Free; premium subscription from $11.95/month |
| **Website:** | https://www.lumosity.com |

**Table 2.2.** Lumosity details

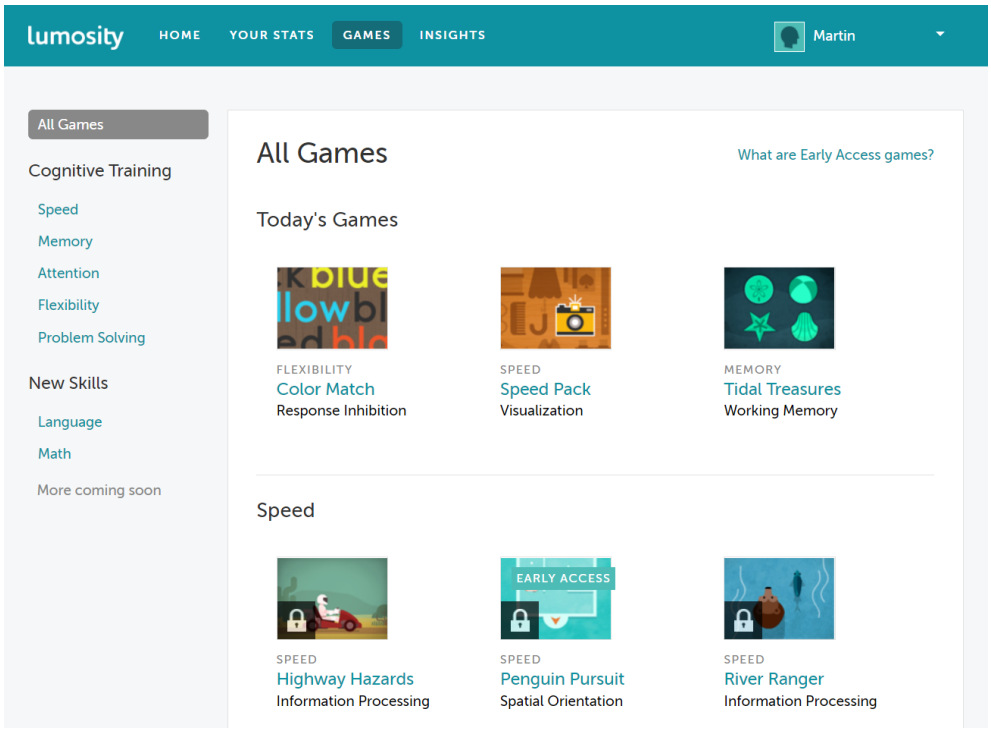



**Figure 2.2.** Screenshot of Lumosity web browser version

Lumosity is well known program for cognitive training designed to improve memory, attention, flexibility and problem solving. Since 2007 when it was launched it had grown its user base to several millions offering more than 60 games and localization to seven world languages. Lumosity claims that it cooperates with more than 40 university researchers worldwide and its training program is used by more than 85 million worldwide.

Lumosity is commercial project offering monthly, yearly, two year plan and free 14 day trial as well. It offers several training plans, daily workouts and detailed insights offering deeper understanding of current training. Games are offered in the following categories: speed, memory, attention, flexibility, problem solving, language, math, mindfulness. It comes in a form of web application and mobile application for iOS and Android as well.

### 2.1.3 Elevate

| | |
|---|---|
| **Author:** | Elevate Labs |
| **Platforms:** | Android, iOS |
| **Price:** | Free; premium subscription from $5/month |
| **Website:** | https://www.elevateapp.com/ |

**Table 2.3.** Elevate details

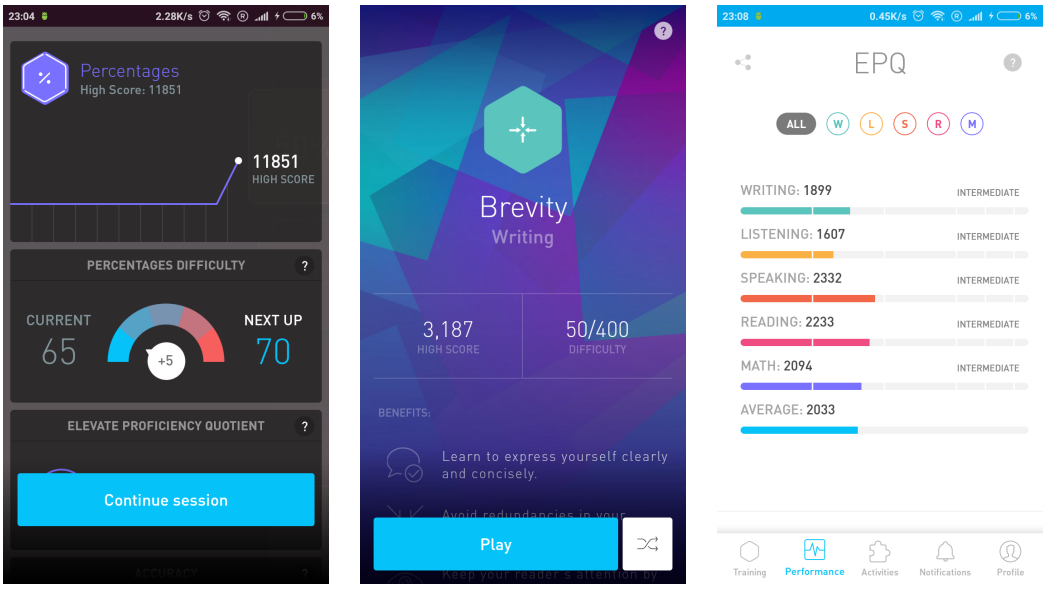



**Figure 2.3.** Screenshots of Elevate

Elevate is a established application focusing to improve attention, speaking skills, processing speed, memory and math skills. It is available as mobile application only, providing multiple personalized training programs that adapt progressively in order to ameliorate training effectiveness. Elevate offers over 40 games in the following categories: writing, listening, speaking, reading, math.

In addition, Elevate offers detailed performance tracking and daily workouts with workout calendar for tracking past results. Furthermore, it contains achievements that motivate the user to progress with trainings. The application is easy to use, providing smooth user experience and having and an attractive colourful look.

| | |
|---|---|
| **Author:** | Peaklabs |
| **Platforms:** | Android, iOS |
| **Price:** | Free; premium subscription from 2€/month |
| **Website:** | http://www.peak.net/ |

**Table 2.4.** Peak details

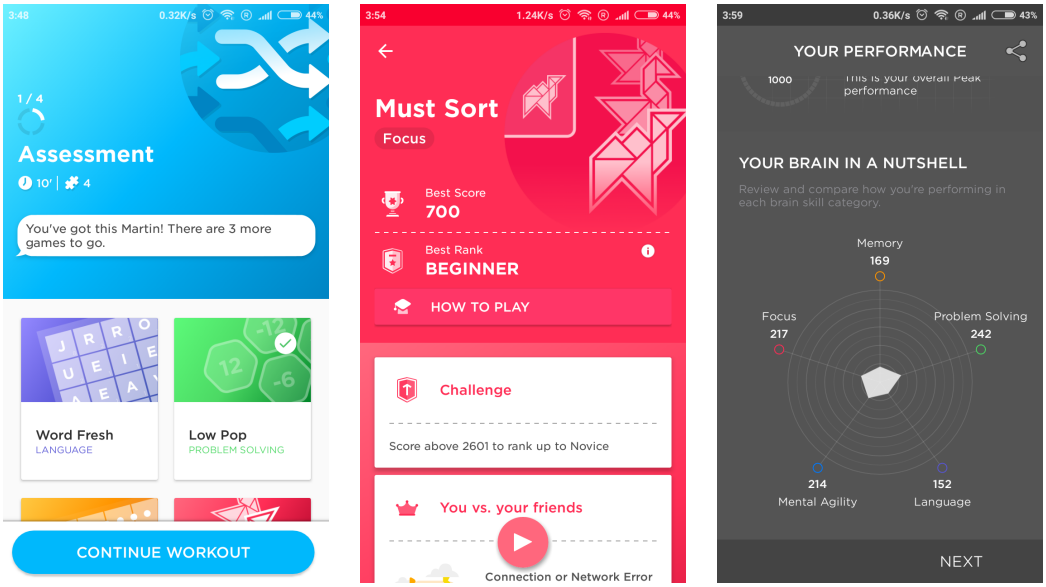



**Figure 2.4.** Screenshots of Peak

### 2.1.4 Peak

Peak is another notable application backed by research of neuroscientists from several worlds top universities. Peak is free to use, but offers subscriptions with monthly or yearly payments, that enables personalized workouts and access to all games. Peak offers over 40 games that come in the following categories: memory, problem solving, focus, mental agility, emotion and coordination. The customized workouts and transparent performance tracking are available as well.

In addition, it enables to compete with friends by comparing results, training calendar with the option to set notifications for daily trainings. These features serve as a good elements for motivation and enabling the progression in the training.

### 2.1.5 NeuroNation

| | |
|---|---|
| **Author:** | Synaptikon |
| **Platforms:** | Android, iOS, Web Browser |
| **Price:** | Free; premium subscription from 42€/3 months |
| **Website:** | https://www.neuronation.com/ |

**Table 2.5.** NeuroNation details

NeuroNation is brain training platform developed in cooperation with psychologists of Freie Universität Berlin. It offers training that includes personalized exercises and progress tracking. It offers more than 60 games divided into categories dedicated to different areas of cognition: Numeracy, Language, Reasoning, Memory and Perception.

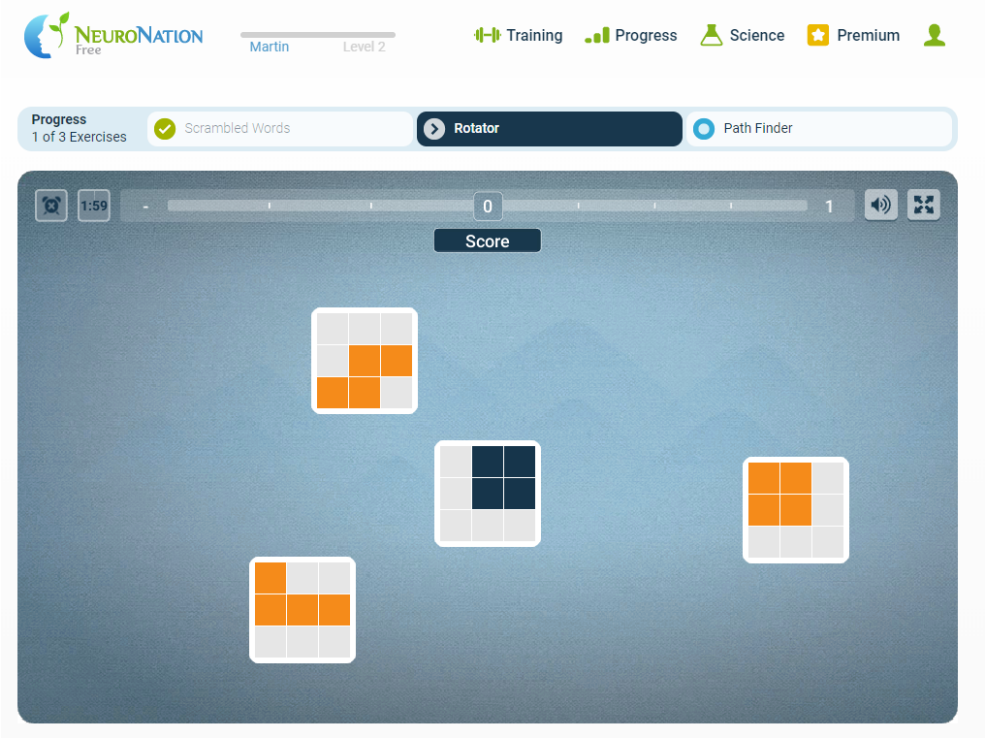



**Figure 2.5.** Screenshot of NeuroNation

NeuroNation offers paid subscriptions for 3 months, 12 months or even Lifetime access and it is localized into four world languages. The most of the functionality unlocks after successful payment paid subscription. The minigames in NeuroNation are very simplistic but most of them have a good gameplay with a flow. Both the mobile and web version contain conservative UI that is not very transparent.

### 2.1.6 Conclusion

Described existing applications are a good source of inspiration for design and development of the application. All of the applications serve similar purpose as developed application but there are several major differences that are as follows:

1. The existing applications except Neurokog represent commercial projects of cognitive training aiming to be used by general public. The developed application is primarily targeted for people suffering cognitive deficits.
2. All of the existing applications contain only simple 2d minigames. The developed application should provide the framework that enables more complex 3d games to be utilised in eg. cognitive training for episodic memory or spatial navigation.
3. The developed application is to utilize 3d graphics and concept of VR, aiming to increase ecological validity of cognitive disorders research and training.
4. Developed application is to contain interface that enables researches of National Institute of Mental Health access to game results of participants.
5. All of the cognitive training applications are backed either, by small team of developers (Neurokog) or the large teams consisting of few tens of developers (Lumosity). Therefore it is necessary to set realistic and fullfilable requirements since the author of the thesis is so far the only software developer of the application.

## 2.2 Personas

The one of the important steps before starting a design phase of the application is to understand the needs of its future users in order to identify the features and functionalities, that are important for them. One of the approaches is to direct the vision about resulting application by creating personas.

Personas are archetypal users of the application that represent the needs of larger groups of users, in terms of their goals, personal characteristics and they act as substitutes for real users to guide decisions about functionality and design. Personas identify the user motivations, expectations and goals and bring users to life by giving them names, personalities and often a photo [1]. The following sections present personas that were created in the interest of directing the development of the application.

### 2.2.1 Persona #1

- **Persona role:** Patient
- **Fictional name:** Adriana Elgasova
- **Job title:** Senior Estate agent

- **Demographics:** Adriana is 59 years old, married, mother of two grown children, she has university degree in economics and she lives in the capital city in the household husband.
- **Goals and tasks:** Adriana was diagnosed Mild cognitive impairment (MCI) and she is attending medical treatment program in the local clinical facility close to her home. She is afraid that her MCI could progress to more serious condition as dementia and she is very motivated to undergo effective treatment program, to keep her mind as sharp as possible.

### 2.2.2 Persona #2

- **Persona role:** Patient
- **Fictional name:** Emil Reinholt
- **Job title:** Auto mechanic in local car repair shop

- **Demographics:** Emil is 38 years old, married, father of three children, he has Vocational school education and he lives with his family in the village in one of the poorest regions Czech republic

- **Goals and tasks:** Emil suffered thrombotic stroke one year ago and he had been hospitalized for a few weeks in clinical facility that is more than one hour of travelling distance to his house. He has cognitive deficit and he regularly visits distant clinical facility for cognitive training. He is looking forward to use computer application in order to reduce his frequent visits to distant clinic. He does not have any smartphone, knows how to use computer for internet browsing but he has never played any computer game and does not know what Virtual Reality stands for.

### 2.2.3 Persona #3

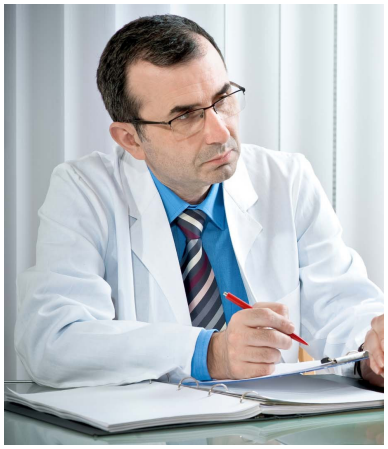
- **Persona role:** Volunteer #1
- **Fictional name:** Karol Vitomir
- **Job title:** Railway Technician at State Railway Transit Company


- **Demographics:** Karol is 29 years old, engaged, no children, he has high school technical education and he lives in the city with her wife.
- **Goals and tasks:** Karol is cheerful and active, he was an voluntary attendant of research program focusing on navigation experiments using Virtual reality and 3d games. He gladly accepted to volunteer to play games using computer application

for cognitive training. He likes to play simple mobile games and after his attendance of navigational experiments, he bought for himself Virtual Reality headset that is compatible with his smartphone.

### 2.2.4 Persona #4

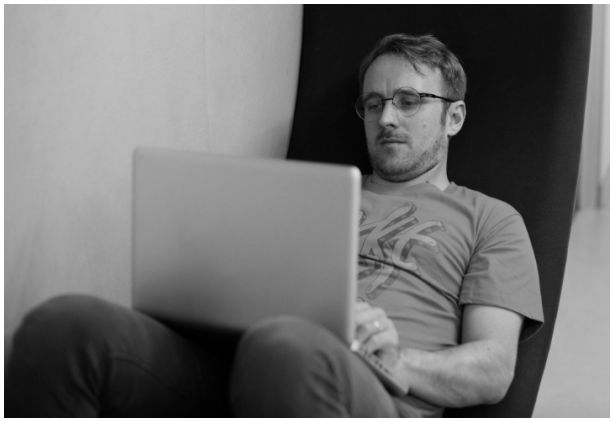
- **Persona role:** Therapist #1
- **Fictional name:** Karl Varlam
- **Job title:** Physician in the clinical facility at department of Cognitive Disorders



- **Demographics:** Karl is 55 years old, graduate of University of medicine
- **Goals and tasks:** Karl, as a physician in the local clinical facility is focused on diagnosing cognitive disorders and creating fitting training games for the patients that are played using paper and pencil. He would like to have a computer application that enables cognitive training for his patients using the computer. He would like to use the application to set up training plan for each patient and see the patients progress.

### 2.2.5 Persona #5

- **Persona role:** Researcher #1
- **Fictional name:** Jozef Oswald
- **Job title:** Postgraduate student in the field of Neuroscience



- **Demographics:** Josef is 29 years old, postgraduate student
- **Goals and tasks:** Jozef is interested in the research of Mild Cognitive Impairment and he would like to use the application for cognitive training to get the data in common data form for analysis. He would like to investigate how the game results of MCI patients compares to game results of standard population.

### 2.2.6 Persona #6

- **Persona role:** Researcher and Admin
- **Fictional name:** Elena Dryagina
- **Job title:** Researcher at National Institute of Mental Health

- **Demographics:** Elena is 41 years old, she is Doctor of Philosophy in the field of Medical Psychology
- **Goals and tasks:** The areas of her scientific expertise are: schizophrenia, cognition and neurofyziology. Elena is part of two research groups and one of her research goals is to study how to adjust cognitive training using the concept of Virtual reality in order to maximize the positive effect of treatments for people suffering various types of schizophrenia. She would like to use the application to obtain game results data for analysis in pursuing of her research goals. She would like to be able to manage user profiles in the application.

### 2.2.7 Conclusion

By employing the concept of personas, and analysing their goals and motivations, it is possible to specify the application user roles by and describing their properties, utilization and acting in the application. The user roles are as follows:

#### Client

Client is a patient that uses the application in contemplation of improving his or her mental health and lowering cognitive deficit. The client uses the application for cognitive training that is adjusted by his supervising therapist and the application provides him the results of his training progress.

#### Volunteer

Volunteer is a healthy person who plays minigames of the application as an enthusiast in order to generate data of gameplay and game results that could be utilized to set the norm for average population. The motivation for volunteer is to play interesting games for cognitive training; he or she could be also motivated by financial compensation by research institution.

#### Therapist

Therapist is typically a physician, the person who supervises clients by setting up their training plans and keeping track of their performance.

#### Researcher

Researcher uses the application in order to have access to game results and expects to be able to export games results to suitable plain text data format for future analysis.

#### Admin

Admin is a role that manages all user accounts of the application. It is a special additional role, that may be assigned to therapist or researcher.

## 2.3 Requirements specification

The proper formulation of the requirements and functionalities that software to be developed has to satisfy is one of the essential steps to be taken in the software development process. These requirements define the purpose of the system, its nature and problems it solves. Author of this thesis formulates these requirements based on the assignment of this thesis as well as consultation with the supervisor and the analysis of the existing applications in 2.1. The application described by this thesis is planned to be subject of the further development and specified requirements reflect this fact.

### 2.3.1 Functional requirements

The following functional requirements define functionalities and specific behaviour that application should be equipped with.

#### 2.3.1.1 User accounts management and authentication

In order to use the application, each user has to have his/her personal account and profile. There is no need for complex authentication system and simple authentication is to be sufficient. Authentication credentials are login name and password. The application allows the admins to deactivate any specific user profile; after deactivation, signing into application using deactivated profile will not be possible (even if password entered during signing in was correct).

#### 2.3.1.2 Registration of user accounts

The registration of new user profile into system will by in competence of users that belong to one of the following roles: therapist, researcher, admin. It will be determined by the following rules:

1. Therapists are allowed to register clients and volunteers, these newly registered users will be supervised by corresponding therapist who registered them,
2. researchers are allowed to register new volunteers that will be supervised by corresponding researcher who registered them

3. admins are allowed to register new users of any role and assign supervisor to them,
4. each newly registered client must have therapist supervisor assigned,
5. each newly registered volunteer must have therapist or researcher supervisor assigned.

#### 2.3.1.3 Minigames

User of any role should be able to list all of the available minigames. Listing should be succinct and listed minigames are to be sorted into the following cognitive categories:

- Visuo-spatial Orientation
- Planning & Decision
- Memory
- Psychomotor & Coordination

Minigames are essential part of the application and serve as a medium for cognitive training. There are four minigames to be implemented and divided into three cognitive categories as follows:

- Sliding puzzle (Visuo-spatial Orientation)
- Sorting game (Planning & Decision)
- Odd even (Planning & Decision)
- Greet (Memory)

Each minigame will use 3d graphics and its design will be subjected for its use as an exercise for one or more specific cognitive functions. The game design of each minigame is to be consulted with the thesis supervisor.

#### 2.3.1.4 Daily training

The application includes daily training routines that help users to train their cognitive skills on daily basis. There is one routine for each cognitive skill category. The Progress of daily training routine for current day and past seven days is visualized and it should motivate the user to progress in training. Training routine triggers minigames that are initialized with parameters which may be specific for each user.

After finishing the gameplay of any minigame, training plan that includes training progress progress would be updated. Access to all minigames for current client profile is to be unlocked after completing daily training routine. The profiles of other user roles have unrestricted access to minigames that does not depend on daily training routine completion.

#### 2.3.1.5 Minigame session results

Each play of any minigame is stored by the application as game session that includes several parameters describing player performance. Some of these parameters are common for all games (eg. start time, minigame round duration) and some are specific for each minigame.

Any researcher profile has access to minigame sessions of all users and can list them for any of the minigames that are implemented. Each game session contains parameters that describe performance of user playing particular minigame at particular time. Format of game session results generally differs depending on each minigame, since each minigame is different.

The application should allow to export all game sessions of any specific minigame as tabular data to external plain text file using Comma-Separated Value (CSV) file format as specified in [2].

#### 2.3.1.6 User profiles overview

The application should allow to list all user profiles that are enrolled in the application. Such a listing should display at least the following information about user profile:

- login
- name
- surname
- user role

In addition, it should provide visual interface to other functionalities operating on user profiles that will be implemented in the future. There are certain particularities that apply to accessibility of user profile listing. These are dependent on the role of user who is signed in and they are as follows:

1. Therapists are allowed to list only client and volunteer profiles they supervise
2. Researchers are allowed to display only profiles of any volunteers they supervise
3. Admins are allowed to display profiles of any user roles

Furthermore, the listing should support the admins to ensure that after deactivation or removal of any therapist profile, they would be able to find clients who were supervised by such a therapist and assign them new therapist. The application should also allow to search user profiles by their login, name, surname and user role.

### 2.3.2 Non-functional requirements

The non-functional requirements specify quality attributes and non-behavioural requirements on the system. It is important to formulate non-functional requirements properly because in case when these conditions are underspecified, the implemented system may be inadequate quality for is intended use. In the other hand when overspecified the system may be too hard to implement. The non-functional conditions described project are defined in the following subsections.

#### Easy to add new minigames

In the future phase of development another minigames will be added. These minigames are being developed by neurologists, game designers and programmers of National Institute of Mental Health and most of the games are being implemented on Unity game development platform. The proposed design of the application should allow to plug in such a games in future and this process should be as straightforward as possible.

#### Easy to port application for mobile and tablet

It should be possible to port the application to mobile and tablet devices. This requirement means that application should be developed in such a way that it is adjustable to various resolutions and input sources. Furthermore minigames have to be designed to use hardware resources effectively so they will run smoothly on mobile hardware.

#### Easy to extend for VR headsets

Minigames should should be extensible to add implementation that enables their use on Virtual Reality headsets using stereoscopic rendering.

#### Easy language localization

The application will be delivered only in English language mutation but in the future, Czech language mutation will be added. This fact should be respected during implementation phase when working with string constants so additional language localization will be hassle free.

#### Data safety

The application will make use of personal data of several clients and employees of medical institutions and therefore safety of such a information must be ensured.

#### Reliability and stability of minigames

The minigames should be reliable and stable in order to be used for cognitive training, comprehending smooth gameplay. Any negative effects that disrupt players experience eg. crashing, freezing and lagging should not be present; minigames should not show any visual or graphical artefacts.

#### Visibility of system status

The application responsiveness to user input should be fluent and user should be informed about its status in situations when actions with longer processing time are triggered.

#### Simple user interface

The application should be easy to use by employing Material Design Guidelines as described in [3].

## 2.4 Use cases and scenarios

Use cases present detailed overview about the usage of application when it is deployed and serve as a starting point for designing of the application. Use cases describe the steps of the user that need to be undertaken in order to use some functionality of the application. Use cases define what the users or roles will be doing in the solution, represents the list of tasks that actors can perform, and is directly related to the requirements of the business process. Use cases are a recognition of the requirements that the project must achieve [4].

There are many use case templates but there is no standard that dictates what properties need to be included in a use case - what needs to be written down for use cases depends on the situation, especially on who is writing it and for what purpose [5]. The following section provides the listing of all use cases categorized by user role and follow-up sections provide detailed description of each use case including scenarios.

### 2.4.1 List of all use cases and scenarios

- Common use cases and scenarios
  - Sign into the application
  - Sign out of the application
  - Display minigames
  - Trigger training routines progress
  - Add new user
  - Display profiles of inactive users
  - Search for the user profiles
- Therapist use cases and scenarios
  - Display profiles of supervised users
  - Filter profiles of supervised users
- Researcher use cases and scenarios

- Display profiles of supervised volunteers
- Display game session results
- Export game session results

- Admin use cases and scenarios
  - List and filter user profiles

### 2.4.2 Common use cases

#### Sign into the application

Sign in interface is shown immediately after application start and signing in is mandatory action for user to take in order to enable use of the application. Login credentials that consist of username and password are authenticated each time and auto sign in option is not present.

1. The application will display sign in screen with fields for login credentials.
2. User enters login credentials and confirms.
3. Application verifies login credentials and in case they are valid, the application logs in the user in case. In case of invalid login credentials or unsuccessful login credentials verification, error message is displayed.

#### Sign out of the application

The functionality to sign out of the application is available for any user who is currently signed in.

1. The User selects `sign out` option.
2. Application signs out user without confirmation and consequently navigates to sign in interface.

#### Display minigames

The user is able to display listing of minigames that are categorized into several categories of cognition.

1. The User selects `display minigames` option.
2. The application displays list of minigames.

#### Display training routines progress

The user is able to display progress of daily training routines.

1. The User selects `display training routines` option.
2. The application displays training routines with corresponding progress

#### Trigger training routines

When displayed training routines, the user is able to trigger any specific training routine.

1. The User selects `start training routines` option.
2. The application triggers training routine of corresponding cognitive category

#### Add new user

The therapist, researcher and admin are allowed to add new user profile to the application.

1. The user selects `display users` option.
2. The application displays available user profiles.
3. The user selects `add new user` option.
4. The application displays dialogue window for adding new user
5. The user fills in required information and confirms.

#### Display profiles of inactive users

The therapist, researcher and admin are allowed to browse through inactive user profiles.

1. The user selects `display users` option.
2. The application displays available user profiles.
3. The user selects `show inactive users` option.
4. The application displays inactive user profiles.

#### Search for the user profiles

The therapist, researcher and admin are able to list user profiles that satisfy text match criteria.

1. The user selects `display users` option.
2. The user activates `search` option and enters text to search.
3. The application displays list of matching user profiles.

### 2.4.3 Therapist use cases

#### Display profiles of supervised users

The therapist is able to browse the profiles of clients and volunteers who are under his/her supervision.

1. The User selects `display users` option.
2. The application displays profiles of users that are under therapist's supervision

#### Filter profiles of supervised users

The therapist is able to filter supervised user profiles depending on their user role

1. The User selects one of the following options:
   - `filter supervised clients`
   - `filter supervised volunteers`
2. The application displays profiles of users that are under therapist's supervision filtered by their role.

### 2.4.4 Researcher use cases

#### Display profiles of supervised volunteers

The researcher is able to browse the profiles of volunteers who are under his/her supervision.

1. The User selects `display users` option.
2. The application displays profiles of volunteers that are under therapist's supervision

#### Display game session results

When minigames listing is displayed, the user is able display game results.

1. The User selects `show game results` option.
2. The application shows game results.

#### Export game session results

The user is able to export game results into CSV file when game results are listed.

1. The User selects `export games results` option.
2. The application exports game results into CSV file located in the applications folder.
3. The application shows the information where is newly created file is located.

### 2.4.5 Admin use cases

#### List and filter user profiles

The admin is able to list profiles of all users and filter them by any various criteria.

1. The admin selects `display users` option.
2. The application displays user profiles of all roles.
3. The user selects one of the following filter options:
   - `filter by particular role`
   - `filter inactive users`
   - `filter with missing therapist`
4. The application displays user profiles based on the particular filter criteria.

# Chapter 3
# Design

This chapter describes the process of application design. The first section presents the application architecture that is followed by the process of GUI and database design. The last section presents the design of minigames.

## 3.1 Application Architecture

One of the most important and challenging tasks in software development process is to design fitting application architecture that accommodates all requirements for consequential steps of software development. Each software architecture is described as the organization of the system that consists of elements and relations among them. It is important to design application architecture properly because changing the application architecture in the future is one of the most expensive changes in development[6].

### 3.1.1 Key Design Principles

There are several key principles to be considered when designing an software architecture. When the principles are applied properly in the process of software development, they should assist to minimize maintenance requirements and promote usability, extensibility and correctness of the software as well. According to [7] and [8] the key principles are as following:

#### Separation of concerns

Each component in the system should be assigned a individual responsibility in pursuance of minimizing interaction points between independent feature sets to achieve high cohesion and low coupling.

#### Single Responsibility principle

Each component should be in charge to only a specific and distinguished feature or functionality.

#### Principle of Last Knowledge

A component should not know about internal details of other components as it helps to avoid the interdependency on internal functionality for a component to achieve better maintainability.

#### Don't repeat yourself (DRY)

Each functionality should be implemented in one component only and it should not be replicated in multiple components.

#### Minimize upfront design

This principle states to avoid big design for whole system and design only the functionality that is necessary.

### 3.1.2 Architecture Design

The non-functional requirements were formulated in 2.3.2 and this section describes how the architecture to be designed is based mainly on the following requirements:

- easy to add new minigames
- easy to port application for mobile and tablet
- easy to extend for VR mobile headsets

There are two distinct architecture design concepts proposed by the author of the thesis. Both of them contain the same design elements but they differ in the interaction and implementation of these elements; they are as follows:

#### User Interface

User interface is part of the application that presents controls of graphical interface to the user, captures user input and triggers actions in application core. In software engineering practice it is sometimes referred to as presentation layer.

#### Application Core

The central part of the application is the application core that accommodates main application logic.

#### Minigames

Each minigame uses hardware accelerated 3d graphics.

#### Database (DB)

The applications has access to data source in form of database that resides on remote database server.

### 3.1.3 Design Concept #1

Design Concept #1 proposes the architecture that consists of

- main application,
- minigames and
- database server.

The Main Application is standalone process that runs on client's device and triggers minigames as separate processes. The Main Application consists of User Interface and Application Core that uses remote database storage.

In order to achieve easy portability of application to multiple platforms and devices, the frameworks, technologies and programming languages that are used for implementation of these architectural elements have to respect this requirement. Fortunately there are many frameworks that enable the application to run on mobile platforms and desktop as well.

The minigames communicate with Application Core via Remote Procedure Call (RPC) or Message-Oriented Middleware (MOM). Most of the RPC or MOM frameworks require that both subjects of the communication use the same programming language, underlying technology or platform but there are several exceptions (eg. ZeroMQ)[1]. When using RPC or MOM on minigames/Application core interface, both sides rely on common communication protocol and each newly added minigame has to respect it.

[1] available at http://zeromq.org/

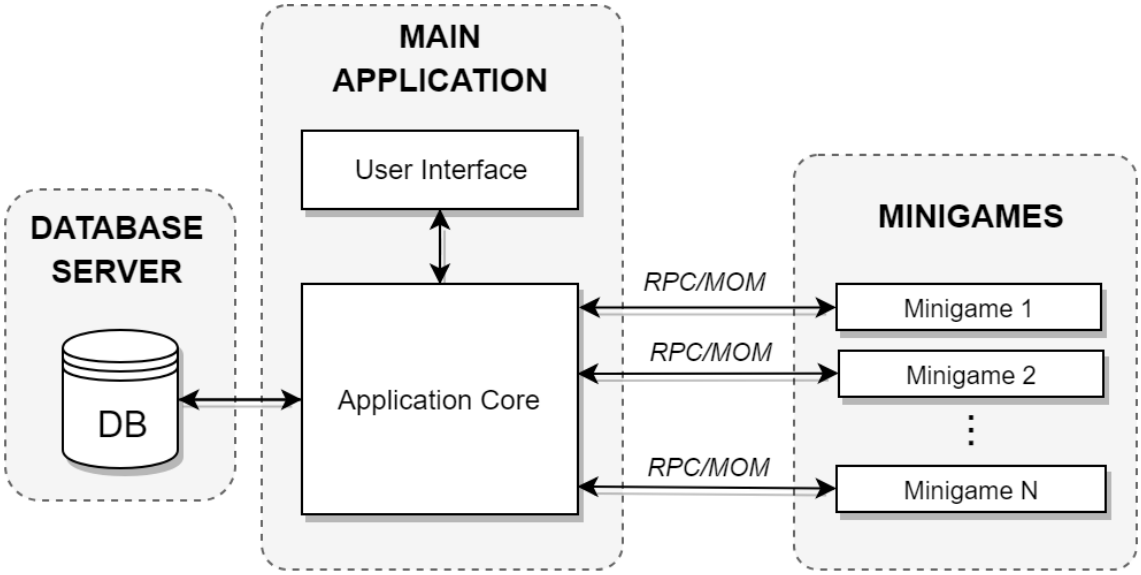



**Figure 3.1.** Design Concept #1

In order to achieve extensibility of minigames to run on VR headsets each minigame has to be developed using framework that facilitates stereoscopic rendering of 3d world that is capable to run on mobile devices.

### 3.1.4 Design Concept #2

Design Concept #2 proposes the architecture that consists of

- Unity Application and
- Database server.

Unity application has a form of monolithic application and standalone process that runs on local target device and makes the use of remote server as database storage. Unity application is built on Unity game development platform using C# programming language and encompasses

- User interface,
- Application Core and
- Database server.

When employing Design Concept #2, the process of adding new minigames is accomplished by merging Unity assets[1] of the main project and minigames as well. Each Unity minigame can be represented as multiple prefabs or gameobjects that may reside in one or multiple scenes. Adding new minigames to the project could be done as following:

1. Adding assets and source code files of minigames into main Unity project (eg. via unitypackage)
2. Recompiling project as into executable standalone application

Porting the application to mobile and tablet can be done by switching build platform in Unity. Since mobile and tablet platforms use touch input, the application has to support it by implementing adequate functionality. Unity development platform provides API for mobile touch input that works across multiple mobile platforms, including Android, iOS and Windows Phone.

[1] Unity assets are various source files including source code that are used in Unity project

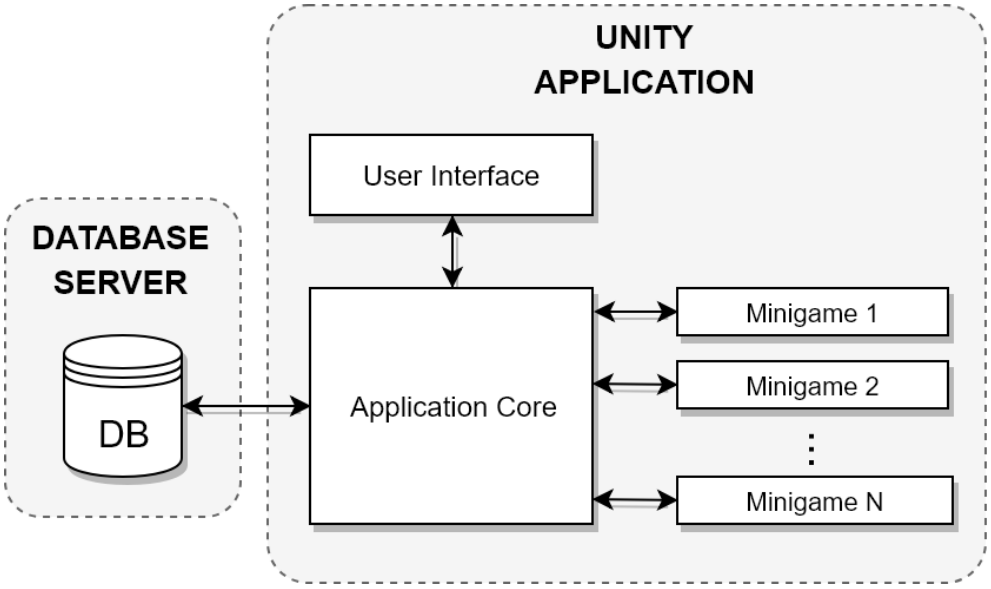



**Figure 3.2.** Design Concept #2

Extending application to be compatible with VR mobile headsets requires to enable stereoscopic rendering for the camera and adjusting user interface of the application. The user interface has to be adjusted in such a way that it reflects the face that user having headset on his head is able to control the application using rotation of his head and pressing only single button that headset contains.

### 3.1.5 Choosing Design Concept

Design Concept #1 provides high degree of freedom for choosing the frameworks and technologies for implementing the Main application and some Minigames. This freedom provides flexibility to choose the most effective, reliable and easy-to-learn software framework to be utilized by the programmer for each component of the application (eg. GUI).

In the other hand, Design Concept #1 brings the requirement to define communication between Application Core and Minigames via RPC or MOM and each newly added game would have to respect it. In addition utilizing RPC or MOM brings the risk to be unable to run such a multitier application using several frameworks and using 3d graphics (possibly with stereoscopic rendering) performance-wise efficiently. Another performance risk is possibility of slow communication of MOM/RPC frameworks. Another downside is that triggering minigames as separate processes is platform specific and this would add another implementational complexity to Application Core.

Design Concept #2 brings the convenience of adding new minigames directly as part of Unity project. Unity development platform supports compiling of the project and deploying on multiple platforms including mobile and desktop operating systems. Furthermore, Unity provides pipeline for enabling the applications to run in VR mode. In pursuance of enabling VR mode and deployment on multiple platforms User interface of the application has to contain functionality that reflects such a use. In the other hand implementing User Interface is laborious because many fundamental elements of user interface (eg. tabs, list views) has to be developed from scratch using 2d graphical elements.

In pursuance of application requirements specification and considering described advantages and disadvantages of each design concept, Design Concept #2 is to be employed in further development.

## 3.2 GUI

Well designed GUI is important in pursuance of developing a user-friendly application. One of the key aspects of successful GUI design is to ensure that application is easy to use and accomplishing some specific task requires as few steps or interactions as possible. There are several principles that are to guide the design process of GUI; some of the principles are formulated in [9] and they are as follows:

#### Clarity and consistency

Those are probably the most important properties of any GUI. Users should be able to quickly recognize interface elements, predict their functionality and use them successfully. Clarity and consistency inspire confidence and lead to further use.

#### Visibility of system status and user control

Users should be kept in control by showing them system status and providing them insight what to expect. The application should not force people into unplanned interactions and surprising outcomes.

#### One primary action per screen

Each screen should contain only single action that is the most important. Screens can contain several secondary actions but they should be kept secondary.

#### Using context

Keep interface controls close to object, the user wants to control.

The resulting application is to be used also by people that have one or more of their cognitive skills weaker than average. This puts the requirement to design corresponding parts of GUI to be simple to use for them. In order to assure that, these parts of the GUI should respect the following general rules:

1. GUI should minimize cognitive load and need to remember things.
2. Interface elements should be large enough and apart from each other with sufficient distance to enable convenience for people with lower skill of coordination and psychomotorics.

### 3.2.1 Task model

Task model is required to capture design knowledge that is utilized in the process of constructing the GUI. It aids to develop the picture and relations of tasks that users are able to perform in the application. It has usually the form of graph that shows cohesion of the tasks.

The Figure 3.3 depicts task model as a graph, based on use cases described in the previous section 2.4. Main tasks that correspond to separate application screens are depicted as rectangles and subtasks as rounded rectangles. Each task contains its name and user roles that are able to perform it. The transition between tasks is shown as a directed arrow and initial task `Login Screen` is highlighted.

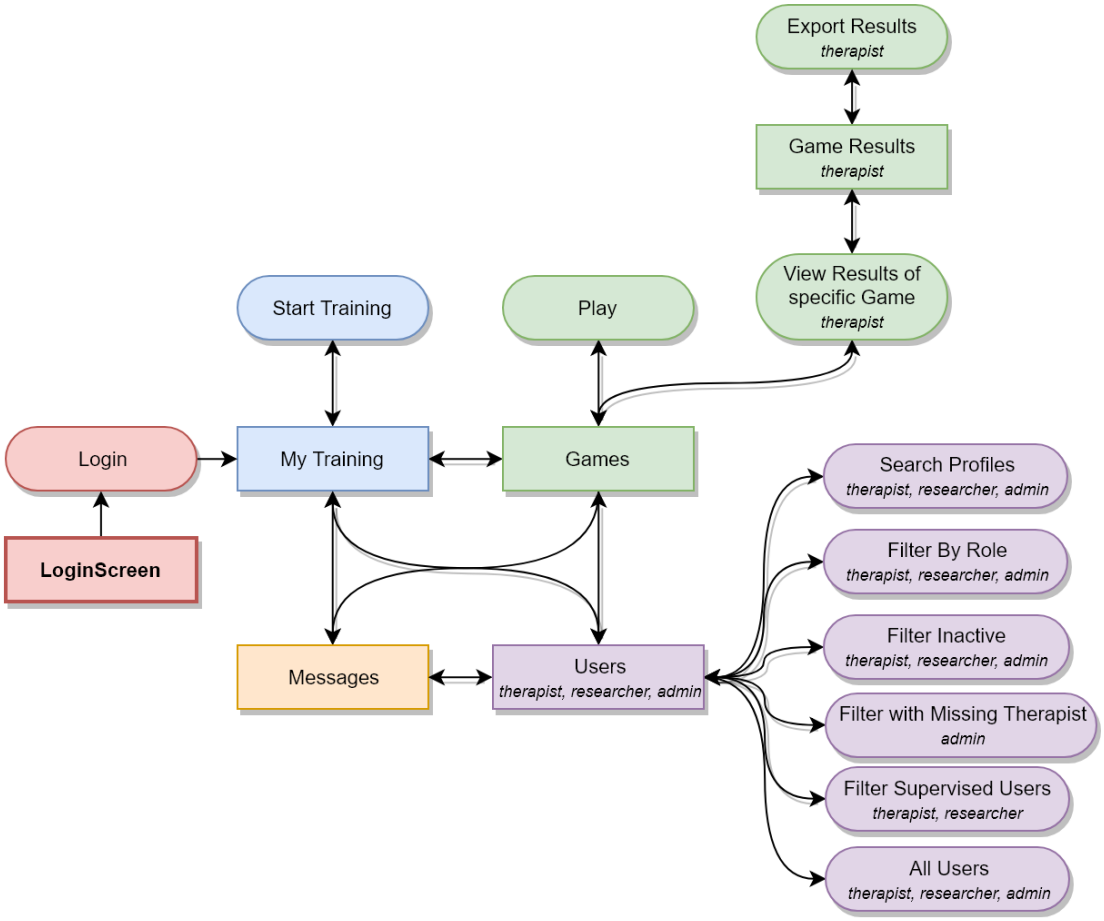



**Figure 3.3.** Task graph of the Application

In addition the GUI of the application is to have standard profile button with icon, displayed in the right to corner. By clicking on this button, the menu with the following three options is displayed:

- `View Profile`
- `Logout`
- `Quit`

The menu is accessible from all application screens. The task `Quit` is required in case when the application runs on desktop in fullscreen therefore window title bar is not shown.

### 3.2.2 Wireframes

Wireframe is blueprint of the interface that depicts allocation and prioritization of content and functionalities available. It is an image where information design, interface design, and navigation design come together to form a unified, cohesive skeleton and serves as a reference for visual design work and implementation [10].

The main function of a wireframe is to provide a hierarchy of information in a design in order to plan a layout that best navigates a user through the information provided for the best understanding and easiest navigation of an interface. The wireframing step is essential, as it allows the designer to determine the layout and user interactions without all the distraction of colors and visuals [11].

In order to keep the the wireframe of the application interface simple and not to depict visual design, the following rules are respected:

- No graphics and no images are used
- No colors are used
- Only default font is used

The wireframes were made by making use of Balsamiq[1]. It is a rapid wireframing and prototyping tool that contains drag-and-drop WYSIWYG editor. As the Figure 3.3 shows, the application consists of six screens and the following sections present single wireframe for each application screen.

#### 3.2.2.1 Login

The login screen consists of interface elements that are necessary for logging into the application. They are located in the center of the screen as is depicted in the Figure 3.4. The `Log In` button visually responds to clicking to let user know that system is authenticating login credentials. In situation when the application is unable to log in the user, error message is shown.

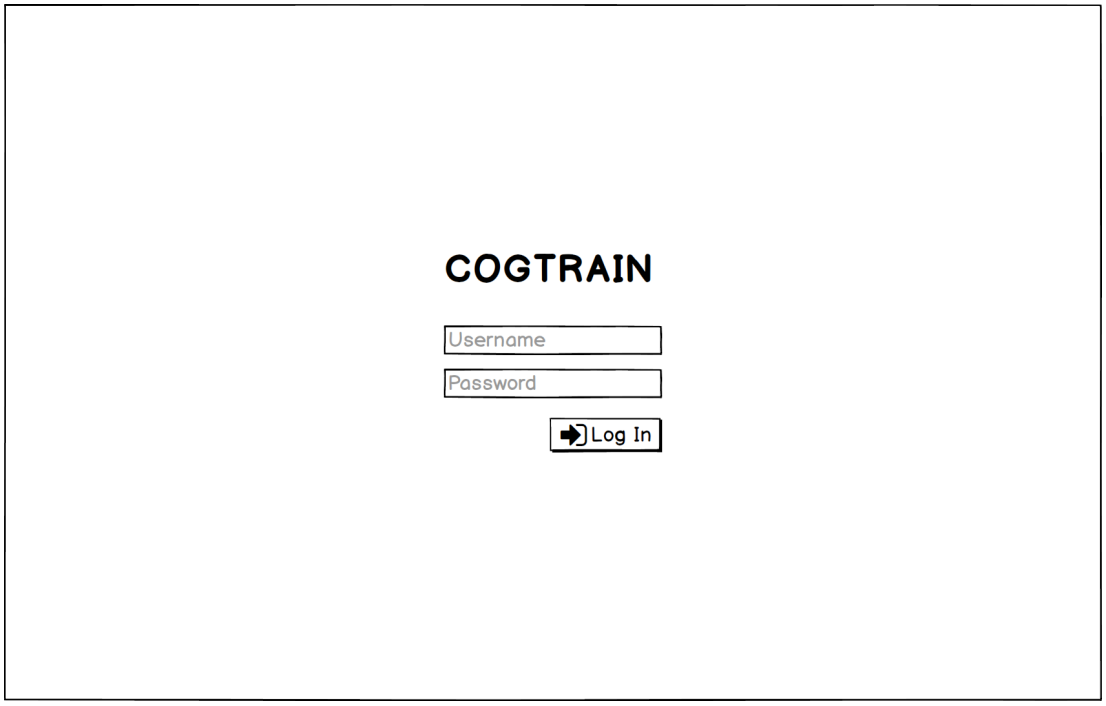



**Figure 3.4.** Login screen

#### 3.2.2.2 My Training

After logging into the application `My Training` screen is shown. This screen is depicted in Figure 3.5. The top bar consists of tab buttons menu for switching to other application screens located on the left and drop-down user menu on the right.

The button `Users` is visible only when current user is logged in using the role of therapist, researcher or admin, therefore it is not visible to the clients nor the volunteers. This feature reflects that only some of the roles are allowed to perform certain tasks that involve user database.

[1] available at https://balsamiq.com/

Main content of the screen contains `Recent Progress` panel, that illustrates completion of the training for each day for past week utilizing bar graph. Other panels illustrate completion of the training per each category of cognition for current day. Every panel of cognition category contains single button that triggers training in corresponding category of cognition.

Some users may be training eg. only one or two cognitive skills and therefore `My Training` screen reflects this by showing only relevant panels.

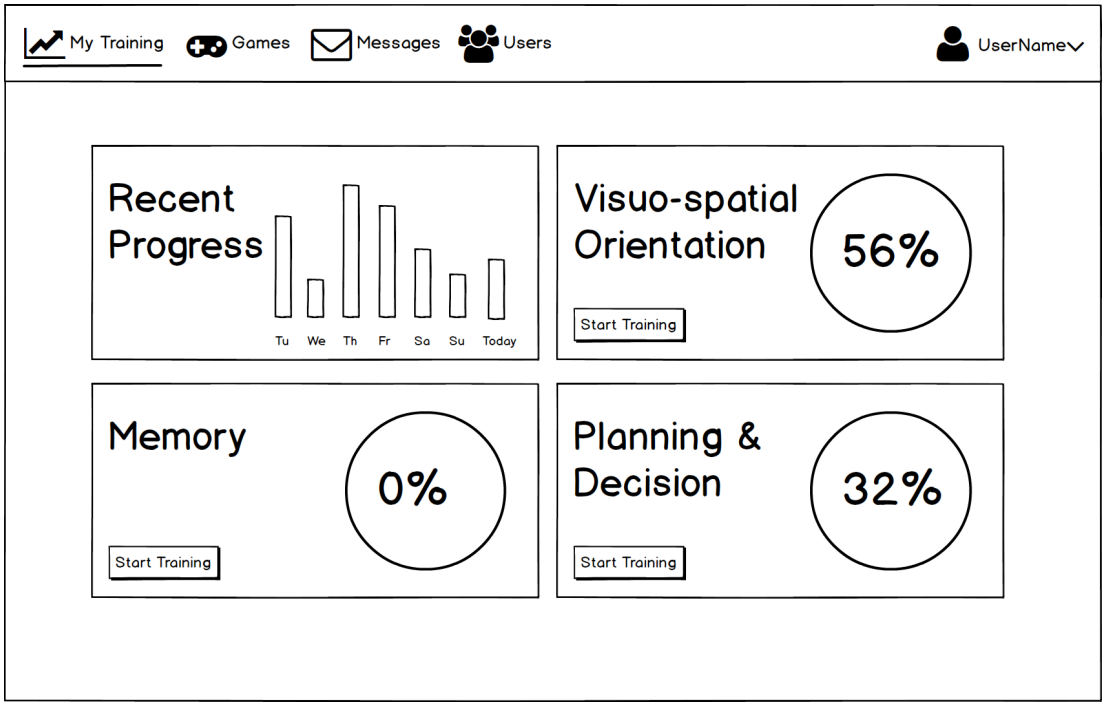



**Figure 3.5.** My Training screen

### 3.2.2.3 Games

The wireframe of Games screen is depicted in Figure 3.6. The Games screen contains four large toggle buttons on the left side, each corresponding to the single category of cognition. By clicking these toggle buttons, the panels that represent minigames are shown or hidden.

Each panel displays to user minigame thumbnail image, minigame name, short description of the minigame and button `Play` that triggers corresponding minigame. In addition when user in the role of researcher is logged in, the panel also contains button `View Results` that navigates to Game Results screen. The screen also contains search bar that allows to filter search minigames.

When user is logged in as a client the application checks if the training for current day is fully completed, ie. user has reached 100% in every category of cognition of user's training plan. In case when user did not finish the training, the application does not allow to play minigames freely until the training has been completed and shows the lock screen as in Figure 3.7.

### 3.2.2.4 Game Results

The Game Results screen is depicted in Figure 3.8. It is accessible only to researchers and it contains table that lists results of minigame sessions for some specific minigame.

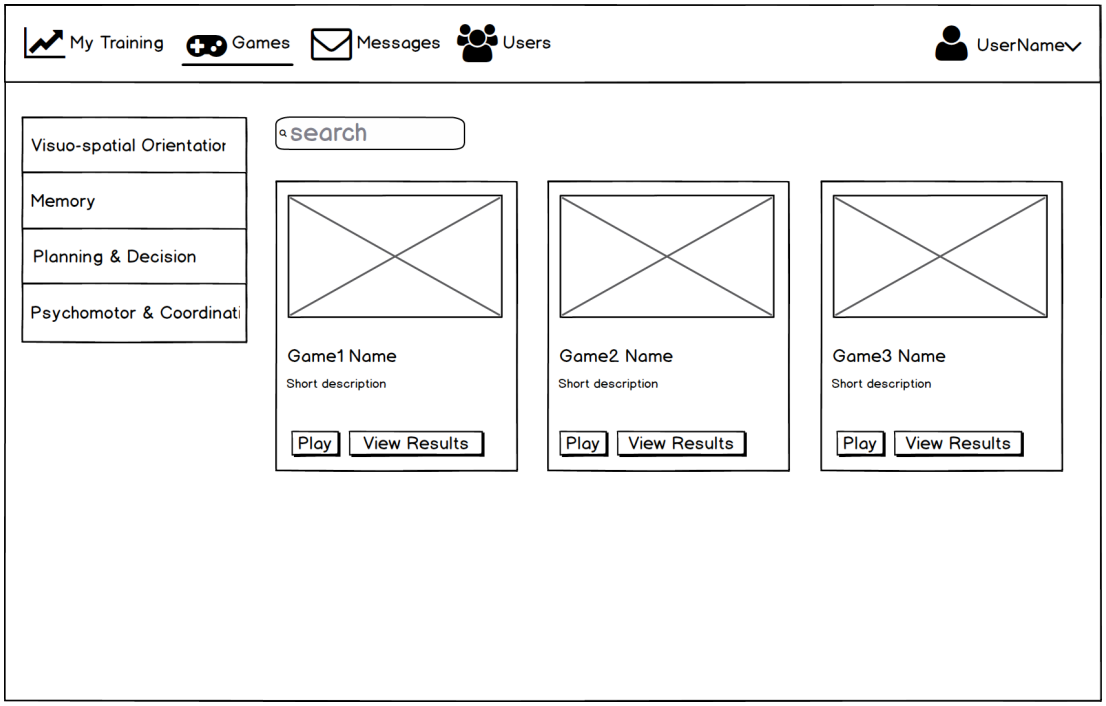



**Figure 3.6.** Games screen

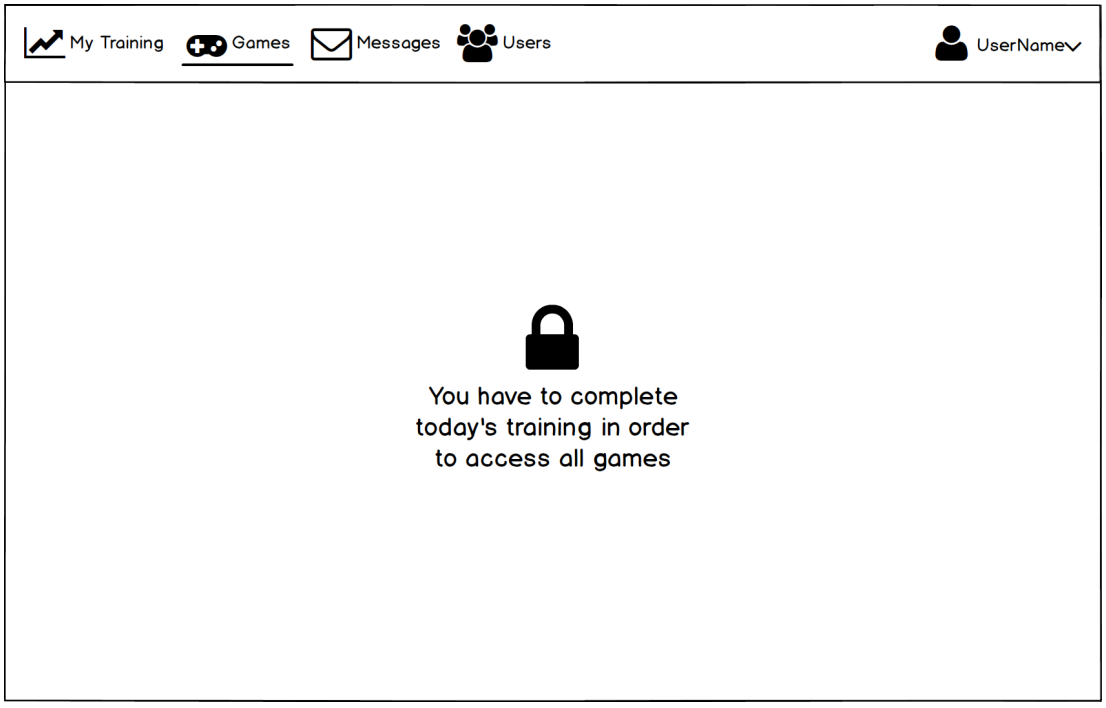



**Figure 3.7.** Locked games screen

Minigame sessions record different parameters for each minigame and so the table may look different for each minigame results.

The screen in addition contains two buttons; one located in top left and second one located in top right. The first one returns back to Games screen and the second one triggers the export of the results to external CSV file.

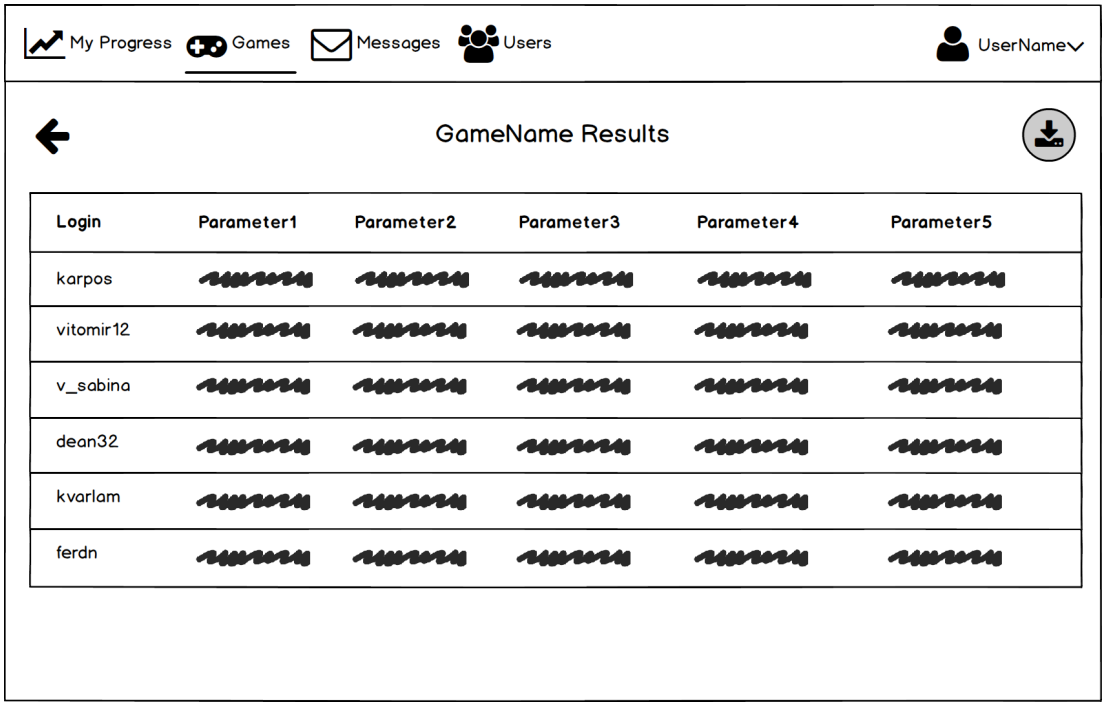



**Figure 3.8.** Results screen

#### 3.2.2.5 Messages screen

The purpose of the Messages screen is for clients or volunteers to contact their supervisors. The Messages screen contain chat interface and it is depicted in Figure 3.9. Left part contains the list of recent conversations and larger part located on the right consists of chat interface. On the top-left, there is filter search filed for past conversations. The top-right part contains + button that creates a new message. Developing of messaging functionality is not part of this thesis.

#### 3.2.2.6 Users screen

The Users screen contains the listing of user profiles and it is accessible only to therapists, researchers and admins. The purpose of the Users screen is for therapists and researchers to have overview of the users they supervise, edit their profiles and trainings. The admin makes use of the Users screen in order to manage the application user profiles. The screen layout and its interface elements differ depending on the role of user who is currently signed in. The wireframe of the screen is in Figure 3.10.

## 3.3 Database

The application needs to store data in order to work properly; as mentioned in the previous section, remote database server is to be utilized in order to satisfy functional requirements of the application. The remote database server provides critical features eg. multiple access control, data redundancy and data security.

### 3.3.1 Choosing database platform

By virtue of the fact that the application is being developed by agile approach, iterative adding new features and extending functionality often brings the need to perform changes in data model. Therefore, as a underlying database, document-based database

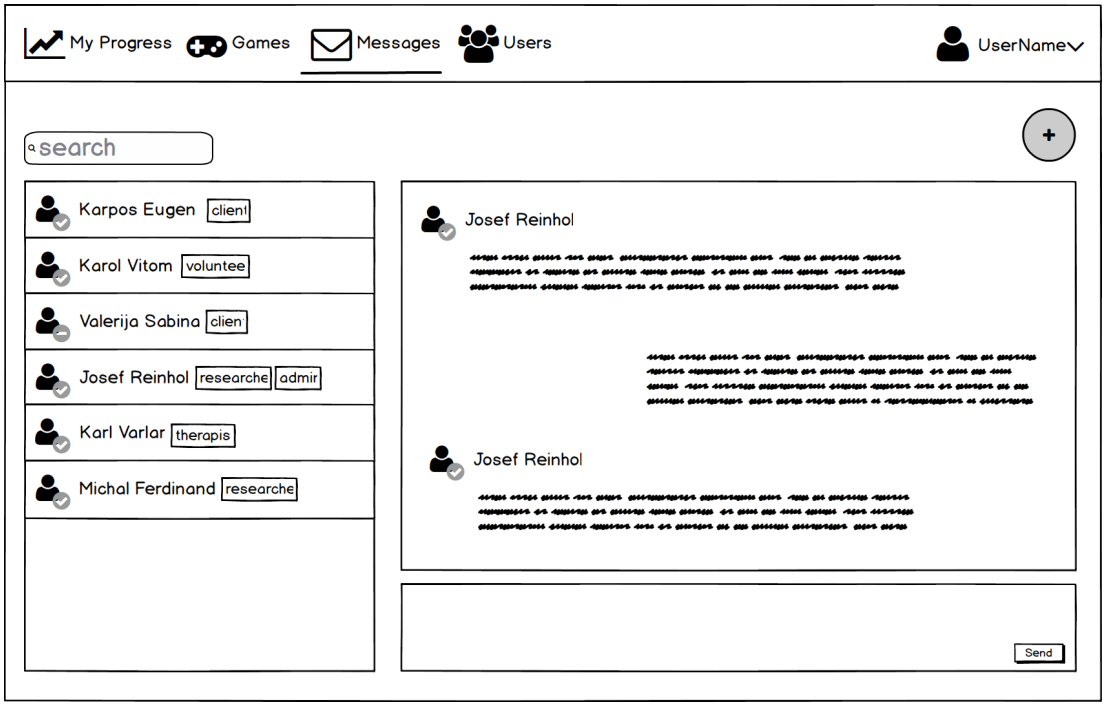



**Figure 3.9.** Messages screen

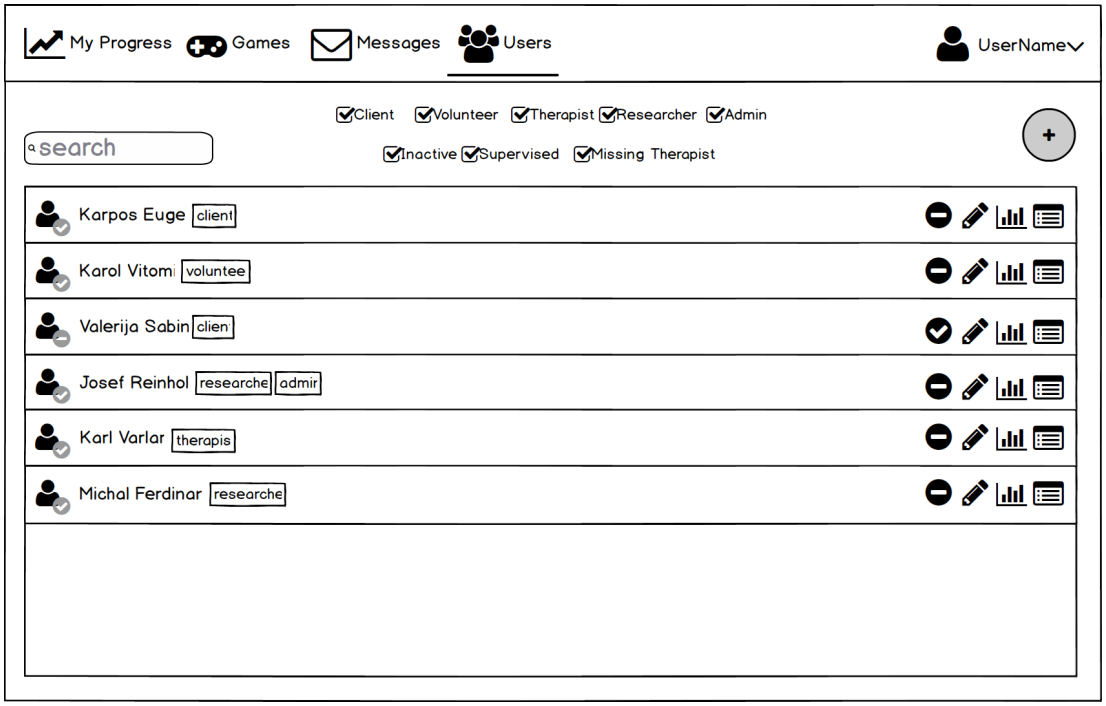



**Figure 3.10.** Users screen

MongoDB was chosen in order to utilize its schema-free concept and flexible documents. The following two instances aim to illustrate the benefits of employing the schema-free concept in the application:

1. When implementing changes related to data for a certain user role, the changes need to be done only in application logic and presentation layer, since document-

based database schema is flexible and therefore it does not pose a slowdown to this modification.
2. Since data structure for each minigame and minigame results differ due to specific nature of each minigame, adding new minigames to the application therefore does not pose a requirement to alter database schema.

In addition to the benefits of schema-free concept, MongoDB's document data model easily maps to the objects in application code. This feature enables to speed up the application development as a result, that programmer does not have to deal with impedance mismatch [12].

### 3.3.2 The aspects of data modelling

Despite the fact that MongoDB database is schema-free and the schema is enforced by the application itself, the process of data modelling is as important as in relational databases. MongoDB stores data in BSON data structure format that is binary variant of well-known JSON, extending it with feature of additional data types and ordered fields. Data modelling for document databases (denoted as document modelling) aims to design BSON data structures that suitably accommodate application's data. The process of document modelling for developed application follows the practices described in [13–14] and [15]; some of the important aspects are summarized as follows:

1. Data must be structured in BSON constructs that are: arrays, objects and values that may be of various data types.
2. Data should be stored in denormalized way, in flat data structures when it fits the application. Emending data in documents may help to reduce the number queries in the application.
3. However, embedding data may pose risk, that some documents may eventually grow very large in size and therefore some queries take too much time and traffic while getting a lots of additional and redundant data.
4. Various applications have different demands from database. Data modelling strategies should reflect this.

### 3.3.3 Document Model

Document model of database is enforced by the application and structure of its classes. The data model is based on task model and the application screens introduced in 3.2.1, while keeping in mind the practices described in previous section 3.3.2. The model aims to minimize queries count (ideally to have one query per user task) and also to minimize size of each transaction even when application data grow in the future. The database is to have the following collections, each collection showing its typical document template; however documents in collection often differ since they are schema-free.

#### 3.3.3.1 User

The collection `User` contains documents with most important profile information about each user.

```
{
    _id : <ObjectId1>,
    Name : <String>,
    Surname : <String>,
    Login : <String>,
    Role : <String>,
```

```
    Admin : <Boolean>
    Supervisor : <ObjectId1>,
    DetailsId : <ObjectId2>,
    Active : <Boolean>,
}
```

#### 3.3.3.2 UserDetails

The collection `UserDetails` contains documents with additional profile information for each user. These data are not embodied in `User` documents in order to minimize size of transferred data, when querying for multiple users.

```
{
    _id : <ObjectId2>,
    Email : <String>,
    Gender : <String>,
    Birthdate : <Date>,
    Education : <String>,
    Occupation : <String>
}
```

#### 3.3.3.3 Game

The collection `Game` contains details for each minigame.

```
{
    _id : <ObjectId3>,
    Name : <String>,
    Category : <String>,
    InTestingPhase : <Boolean>,
    SceneName : <String>
}
```

#### 3.3.3.4 GameSessions

The collection `GameSessions` stores the information related to gameplay of each minigame and game results. Documents in this collection differ, since game results for each minigame to store are different.

```
{
    _id : <ObjectId4>,
    GameId : <ObjectId3>,
    GameName : <String>,
    UserId : <ObjectId1>,
    Start : <Date>,
    Duration : <Integer>,
    <other game session dependent fields>
}
```

#### 3.3.3.5 UserTraining

The `UserTraining` collection contains training data for each user. These data define the parameters and describe the progress of training for the user.

```
{
    _id : <ObjectId5>,
    UserId : <ObjectId1>
    Training : [
        {
```

```
            CogCategory : <String>,
            timePlayed : <Integer>
            timeToPlay : <Integer>
            <other category requirements to fullfill>
        }, {
            CogCategory : <String>,
            timePlayed : <Integer>
            timeToPlay : <Integer>
            <other category requirements to fullfill>
        }

    ],
    LastSixDays : [<Float>, <Float> ...]
}
```

## 3.4 Minigames

The minigames are central part of the application and their purpose is to enable training for the users in various areas of cognition. Some of the minigames have the same or similar mechanic as minigames already implememts

Some of the puzzle minigames have the same game mechanics as minigames implemented in Neurokog.

The gameplay screen of each minigame contains three buttons in top-right corner and their functions are as follows (from left to right):

- `show instructions`; this button shows instructions how to play the game and input controls.
- `skip round`; the button skips current game instance and starts new one. It is for user to press in such a case when he or she is unable to win current game instance.
- `quit minigame`; the button quits the minigame and returns to application's main screen.

### 3.4.1 Sliding Puzzle

Sliding puzzle is well known puzzle game that is older more than a century. The game mechanics is based on moving square puzzles in horizontal or vertical direction in order to establish particular end configuration. The wireframe of the minigame to be implemented is in Figure 3.11. The goal of the minigame is to sort numbered blocks in ascending order, leaving the empty slot in the right-bottom corner. Any numbered block is moved to empty slot after clicking.

The minigame is parametrized by parameter $D_{sp}$ that represents linear size of the numbered blocks; its allowed values are:

- $D_{sp} = 2$ (corresponding to $2^2 - 1 = 3$ blocks)
- $D_{sp} = 3$ (corresponding to $3^2 - 1 = 8$ blocks)
- $D_{sp} = 4$ (corresponding to $4^2 - 1 = 15$ blocks)

Each minigame round results in

- win (after numbered blocks are sorted in ascending order), or
- loss (after `skip round` or `quit minigame` are initiated).

Another parameters that are used are $W_{sp}$ and $L_{sp}$ representing number of consecutive wins and consecutive losses in previous rounds of the minigame. The new value of

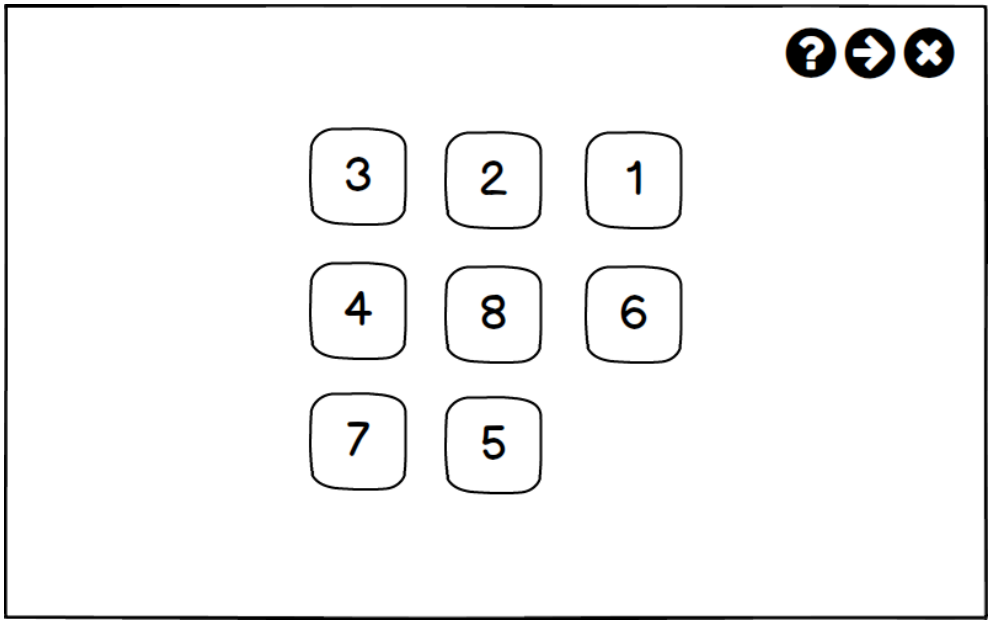



**Figure 3.11.** Sliding Puzzle screen

$D^{\star}_{sp}$ for each gameplay is updated in the beginning of each round and it is determined as:

$$D^{\star}_{sp} = \begin{cases} 3 & \text{for } W_{sp} = 1, D_{sp} = 2, \\ D_{sp} + 1 & \text{for } W_{sp} = 2, D_{sp} < 4, \\ D_{sp} - 1 & \text{for } L_{sp} = 2, D_{sp} > 2. \end{cases}$$

The update function is the same for each patient but it is one of the ways for the therapist to adjust the training for each patient individually. For each value of $D_{sp}$ there is a set of initial states provided by the thesis supervisor and the minigames chooses the initial state from this set randomly.

### 3.4.2 Sorting Game

Sorting Game is another minigame that employs the game mechanics of moving numbered blocks. The goal of the game is to sort the blocks in ascending order (from left to right). The minigame gameplay makes use of single input action; clicking any numbered block $\mathcal{B}$ reverses the order of blocks that reside leftwards from block $\mathcal{B}$, (including $\mathcal{B}$). For instance, clicking on the block labelled 1 in the Figure 3.12 produces sequence 1, 3, 4, 2.

The minigame is parametrized by parameter $D_{sg}$ that represents count of the numbered blocks; its allowed values are: $D_{sg} = 3, 4, 5, 6$.

Each minigame round results in

- win (after numbered blocks are sorted in ascending order), or
- loss (after `skip round` or `quit minigame` are initiated).

The update function for $D^{\star}_{sg}$ makes use of consecutive wins $W_{sg}$ and consecutive loses $L_{sg}$ as follows:

$$D^{\star}_{sg} = \begin{cases} D_{sg} + 1 & \text{for } W_{sg} = 2, D_{sg} < 6, \\ D_{sg} - 1 & \text{for } L_{sg} = 2, D_{sg} > 3. \end{cases}$$

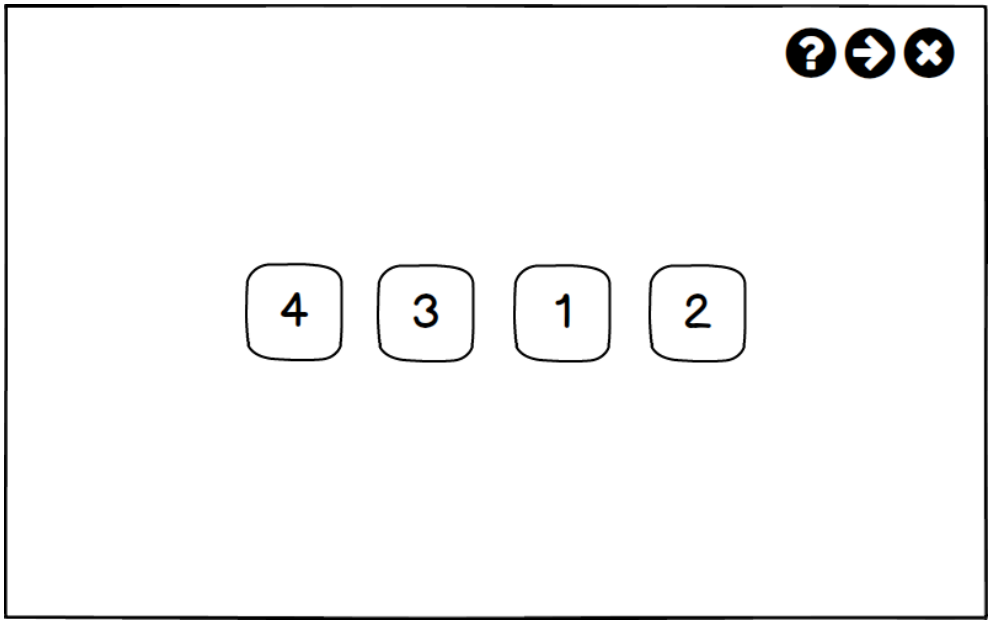



**Figure 3.12.** Sorting Game screen

### 3.4.3 Odd-Even

The gameplay of Odd-Even minigame challenges the user by showing him three numbers and asking if the sum of the numbers is odd or even. The user input action is performed via buttons in the bottom part of the screen. The Figure 3.13 shows the wireframe of gameplay screen.

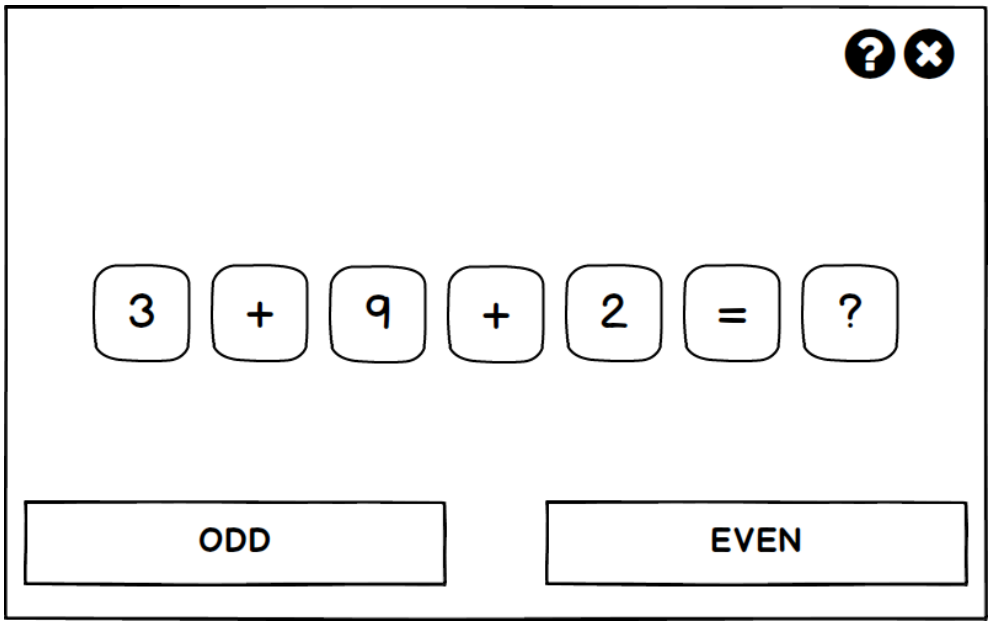



**Figure 3.13.** Odd-Even screen

The minigame round results either as win or loss depending on the correct answer. The game parameter $D_{oe}$ represents the number of digits per number and may have values $D_{oe} = 1, 2, 3$. The update function for $D^{\star}_{oe}$ is as follows:

$$D^{\star}_{oe} = \begin{cases} D_{oe} + 1 & \text{for } W_{oe} = 4, D_{oe} < 3, \\ D_{oe} - 1 & \text{for } L_{oe} = 2, D_{oe} > 1. \end{cases}$$

where $W_{oe}$ and $L_{oe}$ are consecutive wins or consecutive loses respectively.

### 3.4.4 Greet

This minigame sets the gameplay in the 3d enviroment where player moves using first person camera controller and interacts with virtual avatars. The purpose of the game is to train players short-term memory by employing backward memory span. The minigame round consists of two consequent phases as follows:

- **preliminary phase** requests the player to approach virtual avatars consecutively and greet them by action button; the player is asked to remember the order of the avatars in which he greeted them based on their visual appearance
- **test phase** requires the player to approach and greet virtual avatars again but in the reversed order

If the player greets the avatars in test phase in correct order, the round is ended as win; in the other hand, the round ends as loss in the case when the order of greetings is either incorrect or the player cancels the test phase.

The minigame contains two controllers that are used for navigation in virtual 3d space depending on target platform as follows:

- **desktop PC controler** utilizes keyboard and mouse input; as it is common, the keys `w`, `a`, `s`, `d` are used for the movement in horizontal directions in 3d space, the mouse movement is used for camera rotation and the `e` key is used as action button.
- **mobile VR controller** is used as input and navigation method in 3d space. The forward movement of the camera controller is performed by holding VR headset button (ie. long tapping on smarphone screen); the camera rotation employs smartphone gyroscope sensor data and action is performed via double pressing VR headset button (ie. double tap on smartphone screen).

Difficulty of the game is given by count of the persons that the player is required to approach and greet; it is described by the parameter $D_g$; its allowed values are $D_{oe} = 2, 3, \ldots 10$. The update function for $D^{\star}_{g}$ is as follows:

$$D^{\star}_{g} = \begin{cases} D_g + 1 & \text{for } W_g = 1, D_g < 4, \\ D_g + 1 & \text{for } W_g = 2, D_g >= 4, D_g < 10, \\ D_g - 1 & \text{for } L_g = 3, D_g > 3. \end{cases}$$

# Chapter 4
# Implementation

This chapter describes the process of implementation of the application that was designed. Firstly it describes the middleware and support tools that are to be utilized. The latter sections guides through the application and minigames implementation in Unity game development platform as well as creating 3d graphic content.

## 4.1 Development and support tools

A fluent and effective process of application development is backed by proper development and support tools. These tools include programming platform for 3d applications development, programming editor, the tools to create 3d graphic content and tools to manage remote database.

### Game engine

The application is to be build on Unity development platform[1]. It is a cross-platform game engine that enables development of 2d and 3d video games, simulations and visualizations. Unity provides support for recent graphic techniques eg. Screen Space Ambient Oclussion, Parallax Mapping, Dynamic Shadows, Animation Blending, optimized Stereoscopic Rendering for Virtual Reality headsets. Unity makes use of C# programming language and supports building to several platforms including Windows, iOS, Linux and Android.

### Developement Environment

When programming using Unity API, programmers usually rely on Microsoft Visual Studio or Mono Develop. Due to the fact that Microsoft Visual Studio is more future rich by offering advanced smart autocompletion, smart syntax highlighting and many other advanced features, it was selected as by the author of this thesis as IDE to be used.

Visual Studio comes with its own C# compiler but it is unable to compile Unity projects, the reason being the requirement of special Unity C# compiler that is integrated in Unity Editor. In order to achieve functional pipeline for code development, Visual Studio 2017 Tools for Unity were used for maintaining *.sln and *.csproj files. The current state of integration of Visual studio and Unity Editor brings disadvantage of long waiting times when project files are changed outside of Unity Editor and sluggish debugging.

### Unity middleware

Unity game development platform allows to use various types of middleware in order to enable new functionalities of speed up developement. This project makes use of the following Unity middleware:

[1] available at https://unity3d.com/

1. DOTween[1] is very powerful middleware that contains implementation for simplifying transition between various properties of Unity Components eg. position, rotation, color.
2. MaterialUI[2] is middleware that aims to implement Google's Material Design elements for Unity UI system. Unfortunately the asset is outdated and contains a lots of bugs.

### Blender

Blender[3] is well known professional and open-source toolset used for creation of 3d content. It is being in development since January 1995 and offers functionality for entire 3d creation pipeline - modeling, rigging animation, simulation rendering, compositing, motion tracking, video editing and interactive content creation via scripting in Python. Blender is available for multiple for platforms: Windows, Linux and macOS.

Most of the minigames use polygonal 3d models that were modelled in Blender by the author of this thesis. These were exported to Unity using FBX file format with proper settings.

### Makehuman

Makehuman[4] is open-source software for creating 3d models of humanoid characters and it is developed by the community of programmers, artists and academics interested in 3d modelling of characters. Makehuman uses universal, androgynous base mesh of human that can be transformed by tweaking multiple properties eg. gender, age, race. Makehuman allows to tweak more than 1100 properties that interpolate the base mesh into preferred shape. Furthermore, Makehuman allows to attach additional assets to generated humanoid; these include clothes, hair, materials, rigs/skeletons, etc.

### mLab cloud database

As an underlying database system, the database based on MongoDB platform was chosen as most suitable. There are few database service providers that host MongoDB databases; mLab[5] was chosen over MongoDB Inc. due to simple configuration and database management process. During the implementation phase, remote MongoDB server was unavailable multiple times that complicated the process; upgrading mLab database plan would probably solve the issue.

### Studio 3T

Viewing and editing database data and database settings for this project would require to study syntax of several command line tools that are part of the MongoDB installation. In order to minimize the use of command line tools and make development more time effective, software utility with GUI was chosen to be used. There are several such a utilities and Studio 3T[6] was chosen as a best fit for personal requirements of the author of this thesis.

[1] available at http://dotween.demigiant.com/
[2] available at https://materialunity.com/
[3] available at https://www.blender.org/
[4] available at http://www.makehuman.org/
[5] available at https://mlab.com/
[6] available at https://studio3t.com/

## 4.2 Development using Unity

Since Unity does not contain IDE, the project is developed utilizing Unity Editor and Visual Studio as well. Unity Editor is utilized as feature-rich WYSIWYG tool and C# compiler and Visual Studio is used for coding and debugging.

### 4.2.1 Unity project structure

Any project build on Unity platform has the structure as depicted in Figure 4.1. It contains one or multiple scenes, each scene includes render settings, lighting parameters and it is a container for Game Objects. During runtime of any application compiled by Unity, there is a single active scene that controls the render settings and lighting parameters. It is possible to load/unload additional scenes using `SceneManager`.

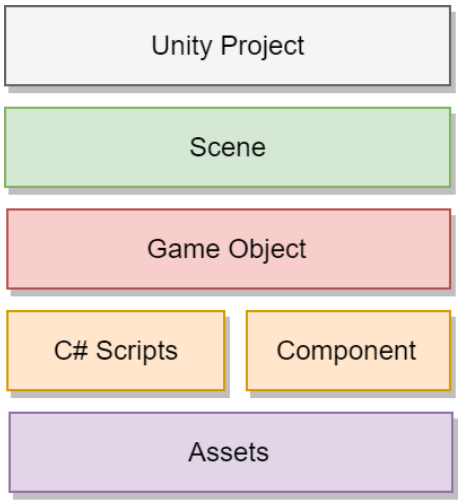



**Figure 4.1.** The structure of any Unity Project

The most important elements of each scene are Game Objects - fundamental objects in Unity. They act as a containers for C# scripts and Components as well. C# scripts and components implement functionality of any Unity application and add properties to Game Objects. Assigning specific combinations of Components with particular parameters and C# scripts to Game Object enables the Game Object to bear particular functionality. Some Components and C# scripts make use of Assets - these are various files containing image data, 3d geometry, etc. The information about Unity project structure, is used to elaborate design of Application Core and User Interface. Architecture and design of database and minigames is described in the posterior sections 3.3 ans 3.4.

### 4.2.2 Programming for Unity

Programming for Unity is done in C# by creating scripts that explicitly derive Unity's `MonoBehaviour` base class. The class contains the following important methods that are implemented in order to define the behaviour of Unity Game Objects and its Components.

- `Awake()` method is used to initialize variables and application state before it starts. It is called once per script instance after all Unity Game Object and Components are initialized.

- `Start()` method is called on the frame when a script is enabled, right before any of the `Update()` methods is called.
- `Update()` is a common function that is called each frame.

## 4.3 Application Core

### 4.3.1 Application Core Internals

The structure of the `Application Core` is depicted in Figure 4.2. It consists of single scene - `Application Core Scene` and this scene contains only one Game Object called `Main Logic`.

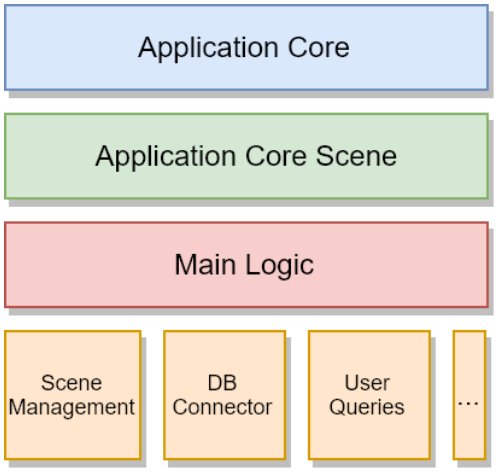



**Figure 4.2.** Application Core Structure in Unity

It is responsible for various functionalities delivered by the C# scripts as follows:

1. `Scene Management` controls loading and unloading scenes that contain various logical subsections of the application. Loading and unloading scenes makes use of `SceneManager` class and utilizes delegates `SceneManager.sceneLoaded` and `SceneManager.sceneUnloaded` that execute initialization and destroy routines for each scene.
2. `DB Connector` initializes and maintains connection to remote MongoDB database by making use of MongoDB .NET Driver. It establishes the connection to the database, authenticates the user and handles connection errors. Furthermore, `DB Connector`
3. `DB Queries` build and trigger database queries. The `DB Queries` persist application data into MongoDB database and supply database data as C# objects to other application components. They also contain definition of enum types that are used by database objects. Database Queries are split into multiple C# scripts based on MongoDB collections eg. `UserQueries`, `GameQueries`, `GameSessionQueries`, `UserTrainingQueries`.
4. Other scripts provide general functionalities utilized mostly by `UI Controllers`.

## 4.4 GUI

The structure of the `User Interface` is depicted in Figure 4.3. It consists of two scenes, representing UI of initial login screen and UI of rest of the application.

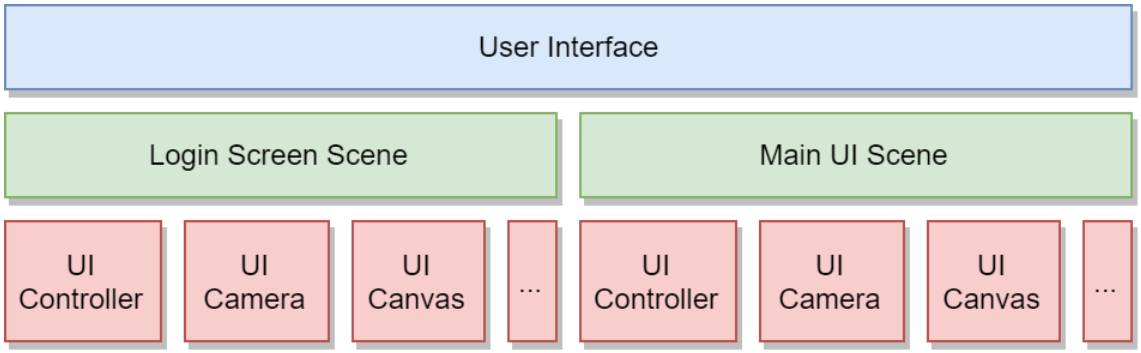



**Figure 4.3.** User Interface Structure in Unity

Login Screen Scene is automatically loaded on the start of the application and after successful user login, the main UI replaces the login screen UI. When user logs out, the loginscreen replaces the main UI of the application. Each of the scenes contains three substantial Game Objects as follows:

1. `UI Canvas` contains Game objects and graphical components compose the UI
2. `UI Controller` ensures interaction of UI Canvas by reacting to user input and providing UI Canvas proper data to update
3. `UI Camera` captures graphical components of `UI Canvas` and renders them to the screen

### 4.4.1 GUI elements

The Game Objects `UI Canvas` and `UI Controller` implement various GUI elements that assemble GUI of the application. In order to achieve clear, fluent and GUI for the application, the GUI elements were constructed by following Google Material Design Guidelines that are comprehensively described in [3].

Implementing GUI in Unity was demanding and time consuming process, since Unity contains only few basic GUI elements with limited functionality. The application makes use of MaterialUI asset, but it does not make the implementation a lot easier, since it is outdated and it is not compatible with Unity native GUI elements and contains visual defects, behavioural abnormalities and compile errors. As a result, most of the GUI elements had to be build from scratch using basic graphical elements and Unity UI components.

The GUI varies depending on user role of current user who is signed in the application. Some of the interface elements and the underlying functionality they control is available only to particular user roles. There is a single GUI interface implemented in the application, but some particular interface elements are hidden depending on the current user role. The script `InitUIComponentForRole` contains the implementation that shows or hides particular interface element; the script is added as component to various Unity Game Objects that implement interface elements and its parameters are set in the Inspector.

## 4.5 Minigames

### 4.5.1 Preparing 3d geometry

The gameplay of all the minigames is set in 3d environment containing 3d models. The minigames: Sliding Puzzle, Sorting Game and Odd-Even (ie. block-based minigames)

contain numbered blocks that are created in Blender; the minigame Greet contains virtual avatars created in Makehuman and the environment is build using models from Urban Construction Pack[1].

#### 4.5.1.1 Modelling of 3d blocks

The decision to make custom numbered blocks in Blender was made after unsuccessful search for appropriate 3d models in several online websites and model repositories eg. Turbosquid[2] and Unity Asset Store[3]. The numbered blocks were created in Blender via polygonal modelling and the process involved the following steps:

1. Standard cube having eight vertexes was inserted into the scene.
2. The cube was scaled along the single axis into rectangular cuboid.
3. Bevel modifier with custom proper setting was applied to cuboid.
4. The meshes of all the numbers and signs that are onto cuboid block were generated from font characters.
5. The material settings were tweaked and the each numbered block model was exported with proper settings as FBX mesh.

The resulting mesh of numbered block in Blender is in Figure 4.4.

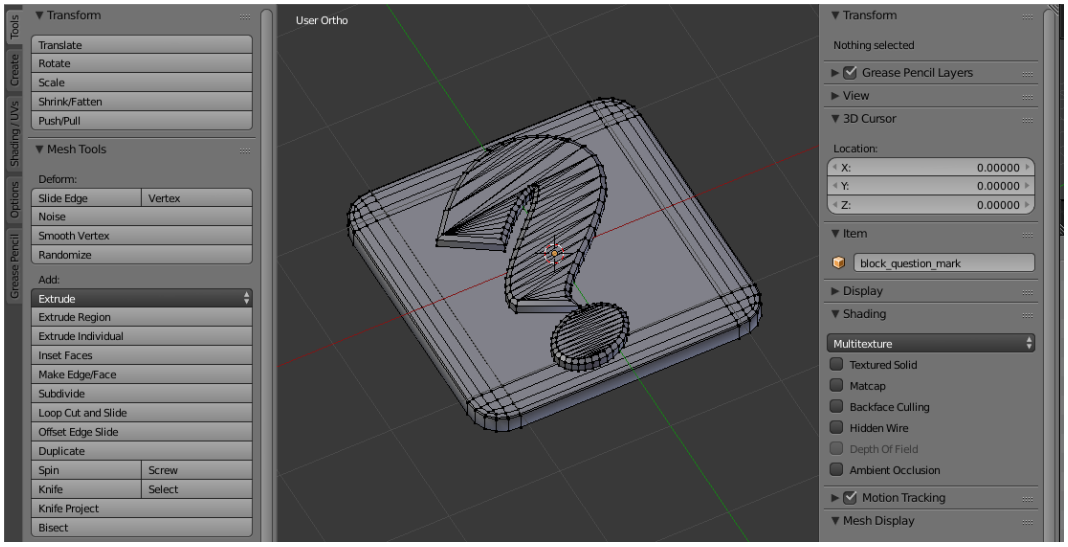



**Figure 4.4.** The mesh of 3d block

#### 4.5.1.2 Generating 3d avatars

The Greet minigame makes use of polygonal 3d models of humanoid characters. There are many such a 3d models available for download but they does not fit most of the implementation criteria that are as follows:

- polygonal count from 10 000 to 20 000
- texture of a size 2048x2048 or similar
- ordinary civil appearance
- common visual style (no specific appearance style eg. cartoon)
- compatible with Unity mecanim as humanoid rig
- free to download and use in the project

[1] available at http://qt-ent.com/UCP/Documentation/
[2] available at https://www.turbosquid.com/
[3] available at https://assetstore.unity.com/

In order to satisfy all the requirements, custom characters were generated by Makehuman. The use of humanoid mesh generator such as Makehuman provides higher level of flexibility to adjust the appearance of the characters in contrast to polygonal editing. In addition, some of the user contributed assets for Makehuman were used as well (ie. clothes assets and body extensions). By utilizing Makehuman, 8 male and 7 female characters were created and exported into Unity via FBX file format. The Figure 4.5 shows one of the male character model.

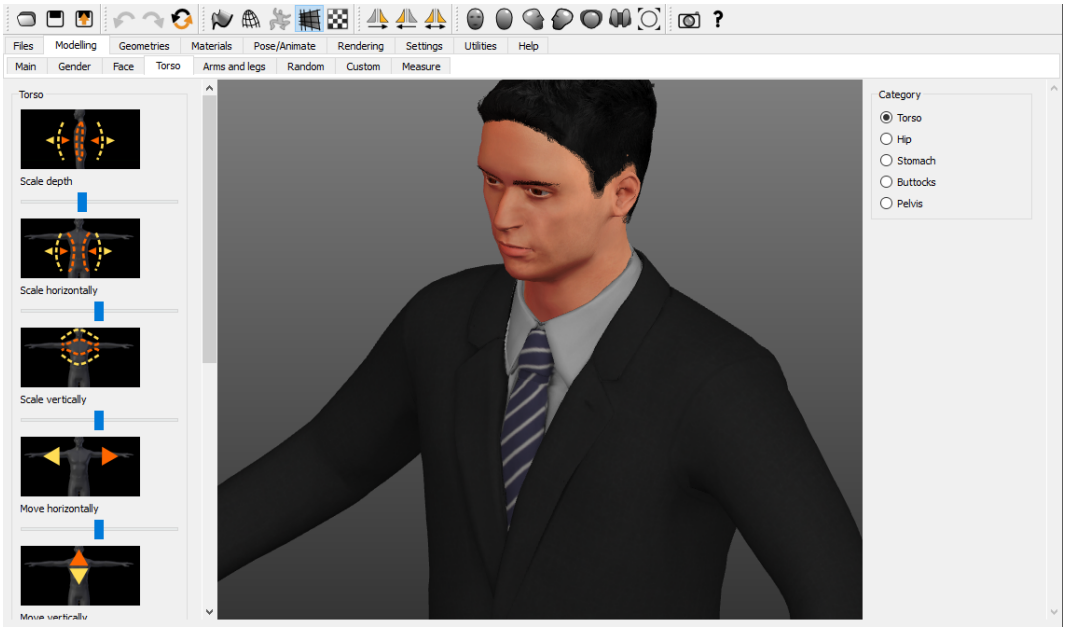



**Figure 4.5.** The resulting 3d avatar in Makehuman

### 4.5.2 Overview of Unity Game Objects

Each minigame is separated into single Unity scene, that is dynamically loaded including all its Game Objects and assets it use, when the minigame is triggered. Separating of minigames into scenes is performance wise and it helps to separate functionalities of each minigames in pursuance of Key Design Principles described in 3.1.1.

The minigame scenes contain multiple Game Objects that make use of various components, C# scripts and assets. The most important Game Objects, according to the author of the thesis, are the following:

- **GAME_LOGIC** is present in each minigame scene and contains most of the C# scripts in each minigames in comparison to other Game Object. It implements core game logic and gameplay mechanics, game environment controllers and game parameter initializers.
- **Directional Light** and **Sun** contain Light Component that defines parameters of lighting for each minigame. Sun is light that simulates sunshine and is used only in Greet game that is set in urban environment; in the other hand, directional light is artificial white light that is used in other three minigames that are set into artificial studio environment. Each minigame uses techniques of Indirect Lighting and Global Illumination.
- **Main Camera** is Game Object that contains C# implementation of camera for each block-based minigame. The camera employs orthographic projection, occlusion culling and multisample anti-aliasing. The camera transitions utilize DOTween

middleware and makes use of **CameraPostions** Game Object as a place-holder of Transforms Objects that are used as 3d world goal positions for camera transition movement between gameplay rounds.

- **BLOCK_PROTOTYPE** is block builder for each block-based minigame. It contains C# implementation that builds Game Objects of custom 3d blocks using 3d meshes and materials created in Blender. Each block is built based upon multiple block properties, eg. color, label, size, position.
- **CHARACTERS_PROTOTYPE** is 3d humanoid builder for Greet minigame that instances desired count of 3d avatars using 3d models created in Makehuman. The avatars are placed in proper (randomized positions) on Navigation mesh with idle animation and proper Rigidbody settings.
- **GAME_CANVAS** contains 2d graphics that form GUI elements and their behaviour. The GUI elements form buttons with actions `show instructions`, `skip round` and `quit minigame` that were described in 3.4.
- **FPS_CONTROLLER** and **FPS_VR_CONTROLLER** are controllers that are part of Greet minigame. They represent Camera movement in 3d environment based upon user input. The controllers functionality is described in 3.4.4.

### 4.5.3 Animations

In pursuance of smooth user experience, the minimages utilize various types of animations acting on different component's properties of particular Game Objects. The minigames Sliding Puzzle and Sorting Game contain animations that act on block properties eg. scale, rotation and they are triggered in various particular situations eg. mouse hovers over the block mesh. Furthermore, the camera movement of block-based minigames is animated when transitiong between minigame rounds. The animations are implemented by utilizing DOTween and various easing functions[1]. Each animation must have defined easing curve, duration, delay and looping as well.

[1] the overview of easing functions is available at https://easings.net/

# Chapter 5
# Testing

This chapter describes the process of testing the application. It starts with performance testing that utilizes Unity Profiler, continues with heuristic evaluation and finally describes usability testing with the users. The outcomes and findings for each test were enabled to improve the application immediately or to note the deficiencies for future development.

## 5.1 Performance testing

Performance testing is very common practice in development of interactive 3d applications. It aims to provide insights, how much processing time is spent by various parts of the application and it enables to analyse the performance of the GPU, CPU and memory. Performance testing assists to find weak performance spots that are to be optimized in order to run the application efficiently and smoothly.

Unity Editor contains profiler that is able to record the performance data of the application. The Unity profiler is depicted in Figure 5.1. The profiling of Unity applications is described in *Advanced Development* that is subsection of the Unity Manual [18]. The key aspect of the application performance profiling is to profile often-used functionalities of the application and analyse the performance spikes. The performance optimizations should be made carefully so they do not cause unexpected application behaviour.


**Figure 5.1.** Performance testing using Unity Profiler

The following sections describe the performance spikes found either in Application Core or Minigames and suggest the eventual code modifications.

### 5.1.1 Application Core

The application core shows short CPU performance spikes in the profiler, immediately after clicking on tab buttons `Games`, `Users` and also when performing filter and search operations in Users section. These performance spikes are caused when data are being transferred from remote database and are shown to the user; therefore the spikes are expectable.

The spikes are not caused by redrawing 2d GUI elements when database data are obtained; they are related only to database queries. The database schema is designed to accommodate application GUI structure by executing single database query per UI action so there is not much room for database schema optimization. The most performance consuming is interactive text search that searches the user profiles in database by executing the function `UserQueries.GetUserSearchResults()`. The author of the thesis considers the following potential optimizations as the most suitable.

1. Reducing search queries count by making search less interactive.
2. Optimizing search queries on the MongoDB server side by adding indexed array of keywords to each document in `User` collection. The array would contain keywords and prefixes of keywords. For example, for user `Martin` it would contain at least the following strings: `m`, `ma`, `mar`, `mart`, `marti`, `martin`.

### 5.1.2 Minigames

All of the implemented minigames were tested in Unity profiler and each minigame shows relatively stable performance. Triggering and exiting each minigame requires extra CPU, GPU resources for loading or unloading game scenes and game objects into memory but the use experience is fluid so performance optimization is not needed.

However, the block based minigames suffer a very short, few millisecond lag after finishing each round of the game. According to the profiler, the lag is caused when minigames are sending game results to Application Core in order to be stored to database. This operation blocks the gameplay precesses until data are stored into database. The suggested optimization would be to store data to DB in parallel, eg. by employing the concept of Unity event system.

## 5.2 Heuristic testing

Heuristic evaluation is very common method that enables software developer or any other software specialist to analyse usability of application's user interface. The method analyses and evaluates validity of ten usability rules within the context of the application's UI and aims to find its deficiencies.

Applying the method is cheaper and less time-consuming than UI usability testing with users. In addition heuristic evaluation enables to find UI deficiencies that participants of usability testing could encounter but they are unable to formulate them. Heuristic evaluation of the application presented in the following section follows heuristics created by Jacob Nielsen, specified in [19] and [20].

### 5.2.1 Evaluation

#### Visibility of system status

+ Gameplay of each minigame is smooth without freezing or lagging
− Some of the database operations cause little lagging and could be done in parallel
− Buttons in minigames located on top-right corner does not provide visual feedback after clicking

#### Match between system and the real world

+ Icons of actions and objects match their real world representation
+ Game results listings for researchers shows parameters with suitable labels in correct order
− Specific icons for each role (Client, Volunteer, Therapist, Researcher, Admin) would help

#### User control and freedom

− Missing logout confirmation
− Missing application quit confirmation

#### Consistency and standards

+ Material Design Guidelines followed

#### Error prevention

+ When adding new user to the system, E-mail field is validated
− When adding new user to the system, it is not obvious, that filling in all the fields is mandatory
− When adding new user to the system, the Age is validated as number, but age min and max limit should be enforced

#### Recognition rather than recall

+ Minigame panels contain thumbnails of each minigame
+ User listing contains icons that are coloured depending on first letter of each user login

#### Aesthetic and minimalist design

+ Only relevant and important information is show for each user role

#### Help and documentation

+ Minigames show play instructions
− Contextual help is not available

### 5.2.2 Conclusion

Heuristic evaluation revealed several deficiencies and provided information that directs the future development of the application. The deficiencies concerning database access were discovered before, during performance testing and few bugs were already noted by the author beforehand. The heuristic evaluation provides useful categorized summary of positive and negative findings. A couple of negative findings were fixed immediately after the evaluation.

## 5.3 Usability testing

Usability testing is valuable step in software development that provides evaluation of the software usability. Even thought the application is not finished and there are several functionalities to be implemented in the future, usability testing provides qualitative insights how to improve the design and direct future development.

The application was tested with both, the users and thesis supervisor. Testing with users was directed by scenario and focusing on minigames gameplay; the testing with the thesis supervisor was in a form of consultation to provide feedback for bug-fixing and extending application functionality.

### 5.3.1 Scenario

The application had been tested with two testers who were given the following instructions:

1. Log into the application with provided login name and password
2. Complete daily training for Visuo-spatial Oriantation
3. Complete daily training for Memory
4. Complete daily training for Planning & Decision

### 5.3.2 Conclusion

The usability testing provided the following insights:

#### Minigame Greet

1. First Person Controller is in some situations unable to move onto curb from the road that is positioned bellow. The bug was immediately fixed by tuning the threshold step values of the controller.
2. One tester encountered difficulties when controlling the FPS controller via mouse and keyboard; the use of controls seemed unfamiliar to her and mouse sensitivity to high. This is a common problem for people who never played any FPS game. The potential solution would be to play the minigame using VR headset; (some people consider controlling VR version more natural and therefore more suitable). Implementing the setting for changing mouse sensitivity is to be noted as a eventual enhancement.
3. Some of the 3d virtual avatars were slightly rotated in horizontal axes. The bug was fixed immediately.

#### All minigames

1. When playing minigames, some gameplay state information should be shown (eg. current difficulty, time elapsed)

### Application Core

1. `Tab` key does not work on the application logic screen. Pressing `Tab` key should pass the focus to next UI element. This deficiency was fixed after testing.
2. The profile button located in the top-right corner of the application should also contain label displaying user role.
3. The tester incorrectly assumed that panels in `My Training` section are clickable and intended to trigger training by clicking on the panel. The solution could be to make the button `Start Training` more visible/larger or make panels for training clickable to trigger trainings.
4. Vertical scrolling in `Games` screen is not sensitive enough. This deficiency was after testing.
5. Game results presented to researchers identify users based on their login names in order to achieve anonymity. The login name often suffices to identity a user since it is similar to persons name or surname. Anonymize the game results so users are not identifiable.

# Chapter 6
# Conclusion

This thesis describes the process of designing and developing the system for cognitive training in a form of a desktop application. The application contains four minigames with 3d graphics that are designed to by utilized in cognitive training; furthermore the application provides basic functionality for daily training, user management and access to game results. The architecture and development platform enables to compile the application to other platforms including mobile.

The thesis covers the process of software engineering divided into several stages that are common for application development. Firstly it analyses the existing applications, investigates target users and specifies application requirements and use cases. The next step proposes the application architecture that accommodates the requirements and elaborates its design in areas of User Interface, Data storage and Minigames. The follow-up implementation phase describes the choice of software technologies and implementation process of the application. The next chapter describes the process of usability testing.

## 6.1 Assignment completion

1. This thesis describes the process of designing and developing the application for cognitive training in a form of desktop application containing four minigames with 3d graphics.
   - The application provides predefined modules for training in the various areas of cognition by utilizing minigames.
   - The application manages accounts of various user roles including clients in order to enable daily training routine that may be specific for each user profile.
   - The application maintains the database that stores data of minigames, user profiles and game results for each user profile; minigame results can be exported to CSV file for analysis.
   - The application provides basic visualization of recent training progress for each user role and daily progress in each category of cognition.
2. The thesis describes the process of designing and implementing several minigames that enable daily training routines.
3. The functionality of the application was subject to usability testing.

## 6.2 Further Development

The application that was developed provides a solid framework for future development. The application is currently still in a phase of development that is to be focused on developing additional functionalities for user profiles management, configuration of training routines, analysing game results and implementing chat functionality. The

development of the application is to continue with existing application architecture, as well as development platform and tools. The usability testing provided valuable insight and there are still some deficiencies that need to be fixed.

# References


[1] CALABRIA, Tina. An introduction to personas and how to create them. *Step Two* [online]. Step Two Designs, 2004 [cit. 2018-02-28]. Available from: http://www.steptwo.com.au/papers/kmc_personas/

[2] SHAFRANOVICH, Yakov. *Common Format and MIME Type for Comma-Separated Values (CSV) Files*. Network Working Group, 2005. Available from: https://tools.ietf.org/html/rfc4180

[3] *Material Design Guidelines* [online]. Google, 2015 [cit. 2018-03-06]. Available from: https://material.io/guidelines/

[4] Defining use cases. *IBM Knowledge Center* [online]. 2016 [cit. 2018-03-22]. Available from: https://www.ibm.com/support/knowledgecenter/SSWSR9_11.0.0/com.ibm.pim.dev.doc/p

[5] KETTENIS, Jan. *Getting Started With Use Case Modeling* [online]. Oracle, 2007 [cit. 2018-03-22]. Available from: http://www.oracle.com/technetwork/testcontent/gettingstartedwit 133857.pdf

[6] MCCONNELL, Steve. *Code Complete*. 2nd ed. Redmond, Washington: Microsoft Press, 2004. ISBN 0-7356-1967-0.

[7] Key Principles of Software Architecture. *Microsoft Developer Network* [online]. 2018 [cit. 2018-03-18]. Available from: https://msdn.microsoft.com/en-us/library/ee658124.aspx

[8] RANJAN, ABHINAV. Software Architecture. *VARIABLE-M* [online]. 2013 [cit. 2018-03-18]. Available from: http://variable-m.com/software-architecture-goals-principles-key-considerations/

[9] 7 unbreakable laws of user interface design. *99designs* [online]. 2014 [cit. 2018-03-25]. Available from: https://99designs.com/blog/tips/7-unbreakable-laws-of-user-interface-design/

[10] GARRET, Jesse. *The Elements of User Experience: User-Centered Design for the Web and Beyond*. 2nd ed. USA: Peachpit, 2011. ISBN 13: 978-0-321-68368-7.

[11] STEADFAST. *Steadfastcreative* [online]. 2016 [cit. 2018-04-23]. Available from: https://steadfastcreative.com/low-fidelity-vs-high-fidelity-wireframes/

[12] FOWLER, Martin. *NoSQL Distilled to an hour* [online]. In: . 2013 [cit. 2018-04-06]. Available from: https://www.youtube.com/watch?v=ASiU89Gl0F0

[13] Thinking in Documents: Part 1. *MongoDB* [online]. 2015 [cit. 2018-04-08]. Available from: https://www.mongodb.com/blog/post/thinking-documents-part-1

[14] Thinking in Documents: Part 2. *MongoDB* [online]. 2015 [cit. 2018-04-08]. Available from: https://www.mongodb.com/blog/post/thinking-documents-part-2

[15] CRAWCOUR, Ryan a David MAKOGON. Modeling Data for NoSQL Document Databases. In: *Channel9.msdn.com* [online]. Microsoft, 2016 [cit. 2018-04-06]. Available from: https://channel9.msdn.com/Events/Build/2016/P468

[16] RIVKIN, Stas. *How to MongoDB in C# — Part 1* [online]. In: . OzCode, 2016 [cit. 2018-04-11]. Available from: https://blog.oz-code.com/how-to-mongodb-in-c-part-1/

[17] RIVKIN, Stas. *How to MongoDB in C# — Part 2* [online]. In: . OzCode, 2017 [cit. 2018-04-11]. Available from: https://blog.oz-code.com/how-to-mongodb-in-c-part-2/

[18] *Unity User Manual* [online]. Unity Technologies, 2018 [cit. 2018-04-03]. Available from: https://docs.unity3d.com/Manual/

[19] NIELSEN, Jakob. 10 Usability Heuristics for User Interface Design. *Nielsen Norman Group: Evidence-Based User Experience Research, Training, and Consulting* [online]. 1995 [cit. 2018-04-21]. Available from: https://www.nngroup.com/articles/ten-usability-heuristics/

[20] NEIL, Theresa. Tips for a Great Flex UX: Part 5. *: Principles and Patterns for Rich Interaction* [online]. 2009 [cit. 2018-04-21]. Available from: http://designingwebinterfaces.com/6-tips-for-a-great-flex-ux-part-5

# Appendix A
# Glossary

| | |
|---|---|
| CPU | Central processing unit |
| GPU | Graphics processing unit |
| GUI | Graphical user interface |
| MCI | Mild cognitive impairment |
| MOM | Message-Oriented Middleware |
| PC | Personal computer |
| RPC | Remote procedure call |
| UI | User interface |
| VR | Virtual reality |
| WYSIWYG | What you see is what you get |

# Appendix B
## MicroSD Card Content

```
project-source.....................................project source files
screenshots.............................screenshots of the application
thesis-source.............................source files for this thesis
videos.....................................videos of the application
thesis.pdf.........................................this thesis in pdf
```