School of Electronic Engineering and Computer Science

Final Report

Programme of study:
BSc (Hons) Computer Science

Project Title:
CogniCare – Think. Play. Grow.

Supervisor:
Dr Mahesha Samaratunga

Student Name:
Binula Harryprashanth Madhavan
(210375121)

Final Year
Undergraduate Project 2024/25

Date: 05/05/2025

Queen Mary University of London

Glossary

- **Accessibility** – Implementing solutions to be used by individuals with diverse backgrounds. For example, users with cognitive or visual impairments.
- **Agile Development** – An iterative Software Development Process that puts emphasis on regular collaboration with stakeholders.
- **API (Application Programming Interface)** – Where rules and endpoints are established to allow for communication between components, such as the frontend and backend.
- **Artificial Intelligence (AI)** - Utilisation of machines to complete tasks that involve human intelligence.
- **Authentication** – The Process of verifying an individual's identity by ensuring security when accessing an application.
- **Axios** – A JavaScript library utilised in creating HTTP Requests (such as a GET or POST request) from the frontend of the application to the backend.
- **Backend** – The server-side of the application data is handled, stored and processed using backend frameworks. This is developed using Flask.
- **Bootstrap** – A CSS framework used to make the frontend of the application stylish and responsive.
- **Cognition** – Mental procedures for areas such as attention, knowledge and memory.
- **Cognitive Decline** – Deterioration of cognitive abilities, which takes place due to dementia
- **Component** – A reusable block of code in React.js, utilised to develop sections of the User Interface.
- **CORS (Cross-Origin Resource Sharing)** – Utilised to enable frontend-backend communication when both are run on different origins (localhost:3000 and localhost:5000)
- **CSS (Cascading Style Sheets)** – A styling language used alongside HTML (Hypertext Markup Language) to style HTML elements within the application.
- **Dementia** – A neurogenerative syndrome which results in the overall deterioration of an individual's cognitive function.
- **Digit Span Test** – A cognitive assessment, where users must recall a sequence of numbers that gets progressively longer. Aimed at assessing an individual's short-term memory.
- **Flask** – A lightweight Python framework, utilised to develop the backend and API of the solution.
- **Frontend** – The client-side of the application, which is developed using React.js. This can be viewed and interacted with by the user.
- **HTTPS** – Ensures security by encrypting data that's sent between the browser and server.
- **Jest** & **'babel-jest'** – These are JavaScript testing frameworks, utilised for unit testing. **'babel-jest'** ensures **'Jest'** can read JavaScript code before tests are run.
- **JWT (JSON Web Tokens)** – A secure method for user data communicated between the frontend and backend during login sessions.
- **Paired Associate Learning** – A cognitive assessment utilised to assess an individual's associative memory, where pairs of items are presented to the user and they must match them correctly.
- **PEP 8** – Python coding convention, ensuring written code is clear.
- **React.js** – A JavaScript library utilised for developing a responsive and interactive frontend.
- **'react-bootstrap'** – A React.js library that enables the integration of Bootstrap components.

- **'react-router-dom'** – A React.js library that enables navigation between different pages in the application.
- **REST API** - Backend architecture that allows the utilisation of HTTPS methods (e.g. GET, POST)
- **Self-Ordered Search** – A cognitive assessment utilised to assess an individual's critical thinking abilities, where users must find a target item hidden behind a set of boxes and must avoid selecting previously chosen boxes.
- **Session** – A period where the user is logged into the application.
- **Stimulus** – A visual prompt which is displayed (during the Paired Associate Learning game), which will then trigger a response from the user.
- **Storage** – Process of storing data (such as scores or tokens), which can be stored locally or on the server.
- **Testing** – Process of ensuring that components or code functions appropriately.
- **Unit Testing** – Testing individual components to ensure they function as expected.
- **User Acceptance Testing** – Testing carried out at the final stages of development, where the system is checked by the primary user to see if it functions as intended (medical students and doctors were used for this process to avoid any ethical issues).
- **User-Interface (UI)** – Visual section of the application which is interacted with by the user (e.g. buttons and forms).
- **User Experience (UX)** – The Feeling of overall comfort and ease users will have while utilising the application.

Abstract

For the final year project, a cognitive game-based web application aimed at patients with dementia was developed. Although existing solutions contained cognitive assessments, many proved to fail in providing an engaging and dementia-friendly environment for users. This project aims to address this gap by implementing a dementia-friendly environment that contains fun and engaging cognitive games for users to take part in. The developed solution contains three common cognitive assessments: The Digit Span Test, Paired Associate Learning and Self-Ordered Search. This application was developed using React.js for the frontend and Flask for the backend.

Agile Scrum-based principles were employed in the development process of the application, feedback from medical students and doctors was heavily used to tailor the solution towards the primary user. Each cognitive game implemented targets a specific area of memory, with different logic and progress tracking. The user's scores for each of the games are stored in the Progress page to aid with monitoring overall cognitive performance. Large fonts and buttons, as well as prompts and video tutorials for each game were utilised to enhance user accessibility. Backend APIs were implemented for retrieving recent and highest scores for each game, as well as login and registration details.

Through User Acceptance Testing and feedback from medical students, this solution fulfilled its function in providing an engaging environment for dementia users to take part in cognitive assessments. Further development of this solution could include the combination of video game artificial intelligence with full-stack development, such as implementation of an Artificial Neural Network (ANN) to adjust difficulty of games and monitor user interaction using intelligent agents, or a Multilayer Perceptron (MLP) to recommend games based on the user's past performances. Moreover, more cognitive assessments can be added to the application to aid in assessing a wider range of cognitive domains.

Contents

Figures

Tables

Chapter 1: **Introduction**

1.1 Background

Dementia is classified as a syndrome with no definitive cure and only symptom-based treatment is known to be available for this. Research conducted by the World Health Organisation states that there will be a rise in the cases of dementia by 84 million by the year 2050 (Joon-Ho Shin et al, 2022). In the UK alone, there is estimated to be an increase in 57% of the cases of dementia until the year 2040, where there will be an estimate of 1.2 million cases of individuals suffering from this syndrome. (Livingston et al, 2020)

As a result of this, there has been an increasing amount of anxiety in the general public around developing dementia (Werner P et al, 2020). In a random survey carried out in Australia regarding the worry of developing dementia, 48% of the participants reported being at least a 'little worried' in developing dementia (Kessler et al, 2012). Early diagnosis of dementia can help to mitigate the potential issues that may rise from this syndrome (de Vugt, M.E. et al, 2013). A key way to diagnose dementia is done through cognitive assessments and some examples of these are the Digital Span Test, Self-Ordered Search and Paired Associate Learning. Cognitive Assessments are used to check an individual's cognitive function such as their thought process, which is extremely vital in diagnosing dementia.

1.2 Problem Statement

This syndrome not only has no cure but has created a lot of worry among individuals who are both directly and indirectly affected by this issue. Treatment is currently utilised as a way of addressing the symptoms of dementia, and unfortunately not being utilised for dementia itself.

The utilisation of cognitive assessments is extremely necessary as a method of not only mitigating the chances of developing this syndrome but also as an identifier for those displaying early symptoms of cognitive decline. However, many existing web applications which are used to help with overall cognitive health are not user-friendly especially towards patients suffering from dementia. Additionally, these applications are not visually attractive or engaging, reducing user motivation for collaboration regularly to improve overall cognitive wellbeing.

1.3 Aim

The aim of this project is tailored toward assisting people suffering from dementia. To solve this issue, I will create an application that will consist of several types of tests to assess cognition, as well as act as a game that both patients and individuals anxious about developing dementia can engage with to improve their cognition. Games that will be included in this application are the Digital Span Test, Paired Associates Learning and Self-Ordered Search.

1.4 Objectives

1. I will conduct a Comprehensive Literature Review on how my application may benefit the user:
    1.1. Understand the effect of cognition on individuals suffering from dementia
    1.2. Understand how dementia affects an individual and how to put their needs into the application appropriately
2. I will then conduct a Comprehensive Data Gathering process on how my application can be better tailored towards dementia patients:
    2.1. Understand how the application can be better suited towards the user's needs
3. Gain a perspective of dementia patients in order to understand new ways to integrate certain elements into the application
4. Design a web application that contains cognition assessments to improve the user's cognition, it will include the following pages:
    3.1. A 'Login/Register Page' which displays options for the user to either login or register to create an account
    3.2. A 'Home Page' which displays navigation to the progress page as well as the list of cognition games that the user can attempt
    3.3. A 'Progress Page' which displays the user's progress in each of the cognition games, so that they can see areas in which they need to improve.
    3.4. A 'Game Page' which displays all the games, so that the user can select their desired game to play.

1.5 Research Questions

- What are the benefits of regular cognitive exercise on a patient suffering from dementia?
- What is a reliable and effective way to measure a patient's cognition?
- What are the benefits of using cognitive games in comparison to cognitive therapy?
- How can private data that is collected from the users of the application be stored securely?

1.6 Report Structure

I understand that in order for the application to function effectively, I must have a sound understanding of cognition as well as dementia. As a result of this, I will carry out research on this by going through research papers and journal articles, specifically those from neuroscience and neuropsychology journals and articles. After having done this, I will carry out a comprehensive data gathering process where I will interview doctors and medical students who have experience with dealing with dementia patients to better understand their requirements for this application.

Literature Review:
I will carry out a comprehensive literature review through research of several studies and articles surrounding the subject of dementia and cognition in order to get an in-depth understanding of the area of interest.

I will look into researching the benefits of cognition assessments on dementia patients as well as on individuals who don't suffer from dementia, for the reason that my application will mainly focus on creating a system containing a variety of interactive cognition assessments and games in order for individuals to not only improve their own cognition but also be able to enjoy the process. Additionally, I will look into other existing applications that provide a similar purpose to the one I'm attempting to carry out and get inspiration from some of their features for my application.

Data Gathering – Interviews and Surveys:
After having carried out my literature review, I will conduct a series of interviews as well as surveys on doctors and medical students using the information I gathered from my comprehensive literature review. By carrying this out, I will be able to find out more about the primary user's needs for this application as well as make it feasible to use.

Through this, I will not only be able to gain more of an understanding about how I can get better design and create the application for its purpose, but I can also gain more of an understanding surrounding dementia and its negative effects on day-to-day life.

1.7 Milestones

1. Carry out a literature review surrounding Cognition and gain a strong foundation in the understanding of Dementia and Alzheimer's. Using this knowledge, carry out interviews on doctors and medical students with expertise in this area.

2. List tasks that each desired feature of the application must carry out as well as what users desire the application to carry out.

3. Learn and become proficient in the necessary programming languages, React.js and Python Flask, to build this solution

4. Set up a GitHub version control platform to work on the development of this application

5. Have a functional web application that carries out the desired tasks effectively.

6. Have a report document containing the necessary documentation and information surrounding this project.

Chapter 2: **Literature Review**

A thorough foundational understanding of the issue as well as how my application can act as the solution to this, will need to be gained. A comprehensive literature review will be carried out first to get an insight into dementia and its effects on cognition and on the day-to-day lives of individuals. After having done this, I will look into existing web applications that also aim to tackle this issue and try to make a comparison between each application, highlighting their advantages and disadvantages and take a note of areas I would like to take reference from for my web application.

2.1 Research Problem

2.1.1 Introduction:

Dementia is a syndrome that has significant neurodegenerative effect on an individual's cognitive ability; completely altering their memory, personality and ability to care for themselves (Gale, Acar and Daffner, 2018). In a study conducted by Torian et al, it was found that geriatric patients with dementia were significantly more likely to suffer from fatal illnesses such as sepsis and drug-related illnesses(Torian *et al.*, 1992). Furthermore, these patients were also found to be more vulnerable to mobility and nutrition deficiencies, which contributed to the excessive burden the patients had on healthcare resources.

The following literature review aims to explore the effects that dementia has on individuals and their cognition, and determine the efficacy of strategies employed to detect and reverse cognitive deficits.

2.1.2 Understanding the Problem:

2.1.2.1. Dementia

The cognitive abilities of an individual, namely the cognitive domains will be severely affected as a result of this disease, ranging from day-to-day social interactions to one's ability to use critical thinking. Cognitive domain is a unique area in a person's cognition that conveys a specific cognitive ability, each cognitive domain works together in order to contribute to an individual's overall cognition. Cognitive Domains are made up of several parts; 'Complex Attention', 'Executive Functioning', 'Learning and Memory', 'Language', 'Perceptual-motor' and 'Social Cognition' ((Hugo and Ganguli, 2014a), Table 3). However, regular cognitive assessments can be carried out specifically to target certain Cognitive Domains in an individual and as a result reverse Cognitive Decline. Examples of these tests are the 'Self-Ordered Search', 'Digit Span Test' and 'Paired Associate Learning'.

The Cognitive Domain of 'Complex Attention' revolves around a person's ability to be able to concentrate and remain focused on specific tasks during a given time. This can have an effect on an individual in their day-to-day life, for example: not being able to concentrate on a task such as cleaning or reading, or in a social setting such as not being able to maintain focus in conversations. These issues are one of many that patients suffering from dementia generally have to go through. However, regular use of cognitive assessment; 'The Digit Span Test' can aim to be of assistance in reversing cognitive decline of this domain. This assessment is utilised as a way to measure an individual's

attention and memory by getting them to recall a sequence of digits both forwards and backwards (Choi *et al.*, 2013).

'Executive Functioning' focuses on the individual's ability to carry out tasks that require them to use problem-solving and critical thinking. Decline in this domain can result in patients struggling in handling multiple tasks or with planning or organising objectives that need to be carried out. As a result of this, tests such as the 'Self-Ordered Search' and the 'Digit Span Test' will need to be performed by the individual regularly to ensure that there is no decline in this domain. The 'Self-Ordered Search' assessment works by presenting a list of items to the participant, for example this could be a box, where there is a hidden object inside one of the boxes. The participant must try to find the object hidden inside one of the boxes without selecting the same box in each of their attempts, this assessment will allow the participant to remain focused and monitor each of the choices they have made. Additionally, this assessment can be used to help with assistance in preventing the cognitive decline in the domain of 'Learning and Memory'.

The Cognitive Domain; 'Learning and Memory', aims at the individual's cognitive ability to retain and store both verbal and non-verbal information in the short and long-term memory(Harvey, 2019). Cognitive Decline in this Cognitive Domain could result in the individual's memory being affected in day-to-day tasks, such as forgetting actions that have already been performed or difficulty in remembering recent events. In order to combat this decline, the 'Self-Ordered Search' and the 'Paired Associate Learning' must be completed regularly by the patient. The 'Paired Associate Learning' assessment involves remembering and recalling information presented, for example an individual is presented with two words and must recall both words after a delay.

'Language' focuses on the cognitive ability of an individual to understand language and be able to give an output to it. This Cognitive Domain looks into areas such as comprehension and vocabulary, as well as fluency. However if there is conveyed to be a decline in cognition in the domain, individuals might start portraying changes in communication such as difficulties in forming words, phrases or sentences, or understanding others through conversations(Hugo and Ganguli, 2014b).

The Cognitive Domain of 'Perceptual-Motor' is an individual's ability to understand and recognise visual information that is presented, an example of this domain being used in a day-to-day task would be when an individual is tasked with navigating to a desired destination using a map. Essential assessments to combat decline in this domain would be the 'Self-Ordered Search'. The 'Self-Ordered Search' would require users to remember which areas have been searched already for the object when going to the next attempt.

2.1.2.2. Effect of Dementia on Physical Illnesses

As touched on earlier, individuals diagnosed with dementia are more vulnerable to physical illnesses. Research carried out by Van Dijk et al, found out that the risk in mortality of individuals with dementia is considerably higher than those not affected by this syndrome. One of the significant factors that have an effect on the risk of mortality in dementia patients alongside the decline in cognitive ability is the increased risks associated with other physical illnesses(van Dijk, Dippel and Habbema, 1991). The possibility of developing physical illnesses as a result of dementia can be seen as extremely worrying as not only does this syndrome have a drastic impact on an individual's cognitive health, but can also lead to damaging effects on one's physical wellbeing as well.

2.1.2.3. Effect of Stroke on Dementia

Proceeding from this point, Andersson and his team carried out a study in 2003 where the objective was to discover a relationship between the development of a stroke and dementia. During this investigation, a sample of elderly patients were gathered to carry out this study, overall it was conveyed that 59.3% of the female participants and 32.7% of the male participants had dementia, in addition 21.5% of both of these groups have had experiences with dealing with a stroke (Andersson *et al.*, 2012). The development of a stroke takes place when there is restricted flow of blood to the brain, causing damaging effects to parts of the brain including areas involved in handling cognitive functions. Through the data gathered from this article, it can be seen that a stroke can be a contributor to the development of dementia.

2.1.2.4. Effect of Diabetes on Dementia

Conditions such as diabetes are portrayed to be a contributing factor to an individual getting dementia. Data collected in 2021 showed that around 536.6 million adults, aged between 20 and 79, globally have been diagnosed with diabetes with the number expected to rise to 783.2 million in the year 2045. Diabetes is a serious global health issue, alongside over 10.5% of the global population having this chronic disease, the worldwide cost of dealing with this illness was shown to be $966 million in 2021 with an estimated rise to $1.054 billion by the year 2045 (Sun *et al.*, 2022). A study carried out by Toshibaru Ninomiya to find the relationship between diabetes and dementia, conveyed that the presence of diabetes in an individual can increase their likelihood of developing dementia by 1.5 to 2.5 times. One factor of diabetes that can increase the risk of dementia is high blood pressure, as this can cause issues with blood flow to the brain hence damaging parts of the brain, namely those controlling the cognitive functions, over time. Other factors include Glucose Toxicity, elevated levels of sugar in the bloodstream can lead to an increased likelihood of blockages taking place in blood vessels, resulting in limited blood supply to the brain. Due to this, cells in the brain can die due to deprivation of necessary supplements causing permanent and drastic damage to cognitive functions in the brain (Ninomiya, 2014).

2.1.2.5. Effect of Depression on Dementia

Mental health conditions such as depression are conveyed to be as much of a major contributor as physical illnesses to the development of dementia. Depression increases the chances of developing dementia by 2 to 5 times, with there being an even higher risk for individuals who get depression later in their life. There are several mechanisms that could cause the development of dementia from depression such as Vascular Disease, Hippocampal Atrophy and Glucocorticoid Dysregulation (Byers and Yaffe, 2011). As touched on earlier, an effect of depression is the rise in blood pressure caused by Vascular Diseases, which can have a significant effect on an individual's brain function. This can result in restricted blood flow, causing damage in brain cells which in turn can lead to drastic damages to the cognitive functions of the brain. Hippocampal Atrophy is a degenerative process that takes place as a result of both stress and depression, however other factors such as ageing can have an effect on this. This is the shrinking of the hippocampus, which is a vital area of the brain which controls lots of cognitive functions such as learning and memory. Over time, Hippocampus Atrophy can cause significant cognitive issues such as loss in memory to an individual.

2.1.2.6. Dementia Worry

Many suffer from what is called ‘Dementia Worry’, which is the fear of the development of dementia, and this is mainly common in middle-aged and elderly individuals. Research conducted by Kessler et al, conveyed that out of all participants collected for a survey conducted to find out more about Dementia Worry, 26% of participants stated their fear of dementia over the development of other conditions and among these participants, 39% were over the age of 55 (*Dementia worry: a psychological examination of an*

unexplored phenomenon - PMC, no date). The worry around dementia can cause a lot of anxiety and stress among individuals, especially among those who are older this can have drastic effects, and in worst cases this can lead to issues with depression.

2.1.2.7. Detection

In order for an individual to be diagnosed with dementia, cognitive and neurologic examinations will need to take place, alongside analysis into the patient's medical history (Arvanitakis, Shah and Bennett, 2019).

Cognitive Examinations are utilised to be able to identify any potential cognitive impairments in an individual. An examination that is commonly used for this is the 'Mini-Mental State Exam' (MMSE), which examines all Cognitive Domains so that a score can be provided to better understand the severity of any impairment in the overall cognition of an individual. Additionally, another screening tool used in this examination is the 'Montreal Cognitive Assessment' (MoCA). Though this is similar to MMSE, this assessment is portrayed to be more thorough with its analysis on cognitive impairment as it gives a more detailed analysis of all cognitive domains. A study carried out by Nasreddine et al, conveyed that out of 94 participants the MMSE and MoCA assessments were carried out. Through the MMSE assessment, there showed to be an 18% sensitivity for the detection of Mild Cognitive Impairment (MCI), and the MoCA assessment conveyed there to be a 90% sensitivity (Nasreddine *et al.*, 2005).

Neurologic Examinations however focus mainly on imaging of the brain to identify any abnormalities that may be leading to cognitive impairment. One imaging technique utilised for the identification of dementia is Magnetic Resonance Imaging (MRI), which takes images in high resolution of an individual's brain structure namely detecting changes in specific regions such as the Atrophy or Hippocampus. Both regions are extremely vital areas in charge of the Cognitive Domain of 'Learning and Memory'. Additionally, MRI is used in the detection of white matter lesions in the brain, which is used as a confirmation for vascular dementia in an individual. Another technique used is Positron Emission Tomography (PET), which aids in the detection of dementia by examining the metabolic activity (Raval *et al.*, 2022).

2.2 Existing Solutions

Solution Name	Description	Advantage(s)	Disadvantage(s)	Inspiration
Lumosity Web and Mobile Application	Application used for cognition training with the ability to give exercises tailored to the user's experience	• Allow users the ability to track their progress and also get feedback on their performance on the app	• Paid subscription is needed to be able to access all features of this application	Allows users to assess their cognition in a fun and interactive way
MindMate Web and Mobile Application	Application that not only provides cognition exercises but also provides physical exercise and healthy recipes	• Application enables users to communicate with other users	• Application is not very user-friendly, can be overwhelming especially for patients suffering from dementia • Paid subscription is needed to be able to access all features of this application	Allows users to assess their cognition in a fun and interactive way
Constant Therapy Web and Mobile Application	Application that gives supports not only with cognition but also with speech and language	• Application contains several thousands of exercises for the user to complete	• Application is not very user-friendly, can be overwhelming especially for patients suffering from dementia • Paid subscription is needed to be able to access all features of this application	Allows users to assess their cognition in an interactive way

CogniFit Web and Mobile Application	Application provides cognitive exercises, that ranges from memory exercises to verbal reasoning	• Allow users the ability to track their progress and also get feedback on their performance on the app	• Application is not very user-friendly, can be overwhelming especially for patients suffering from dementia • Paid subscription is needed to be able to access all features of this application	Allows users to assess their cognition in a fun and interactive way

Table 1 - Existing Solutions Table

2.2.1 Lumosity

Lumosity is an online application aimed at assisting individuals to improve their cognitive health over time. Research conducted by Butler et al conveyed the extreme significance of regular cognitive assessments are vital for the early detection of cognitive impairments. In addition, these assessments can be utilised as a way to be able to differentiate between certain types of dementia that individuals may be suffering from (Butler and Katona, 2019). This aligns with the aim of this application, as it provides exercises for each Cognitive Domain such as certain exercises for memory, whilst others pay more attention to problem-solving. Moreover, Lumosity contains a platform for users to be able to track their progress over time and be able to identify areas of improvement.

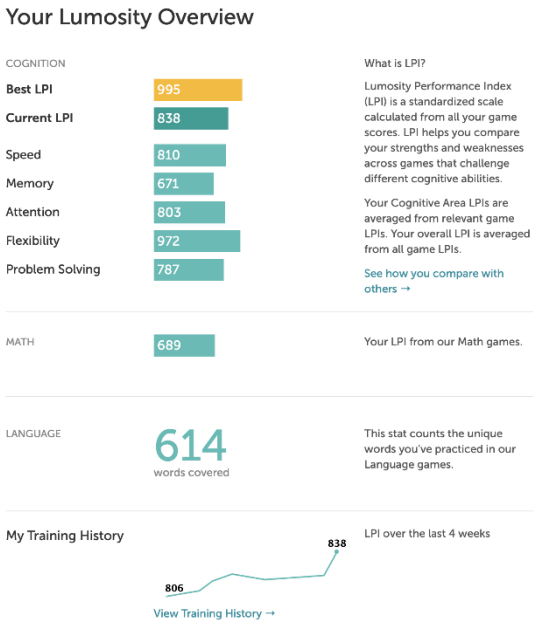


Figure 1 - Screenshot of Lumosity Progress Page, Source = https://www.sitejabber.com/reviews/lumosity.

2.2.2 MindMate

This application utilises all the key features included in Lumosity. This application is said to mainly be aimed at patients suffering from Alzheimer's however it can also be utilised as regular exercises for patients with dementia. This application includes physical exercises for users to stay fit and healthy as well as communication tools for users of the application to connect with caregivers and other users allowing for regular collaboration and social interaction. Research conducted by Piolatto et al, showcased that one important feature to improving cognitive health is regular social interaction which correlates with a reduced rate of cognitive decline in individuals (Piolatto *et al.*, 2022). This application aligns with this as it puts heavy emphasis on regular social interaction.

2.2.1 Constant Therapy

Constant Therapy is an application not only tasked with helping users to improve their overall cognitive health but also with their speech. This aligns with the goals of other existing solutions that have been analysed so far and acts as a way to support dementia patients with their cognitive decline. This application, unlike many others, includes thousands of different cognitive exercises for users to complete. In addition, it gives heavy emphasis on regular cognitive exercise by allowing users to be able to set reminders to help with maintaining a strict schedule on completing these as well as giving users the ability to track their performances on exercises aimed at targeting different Cognitive Domains.

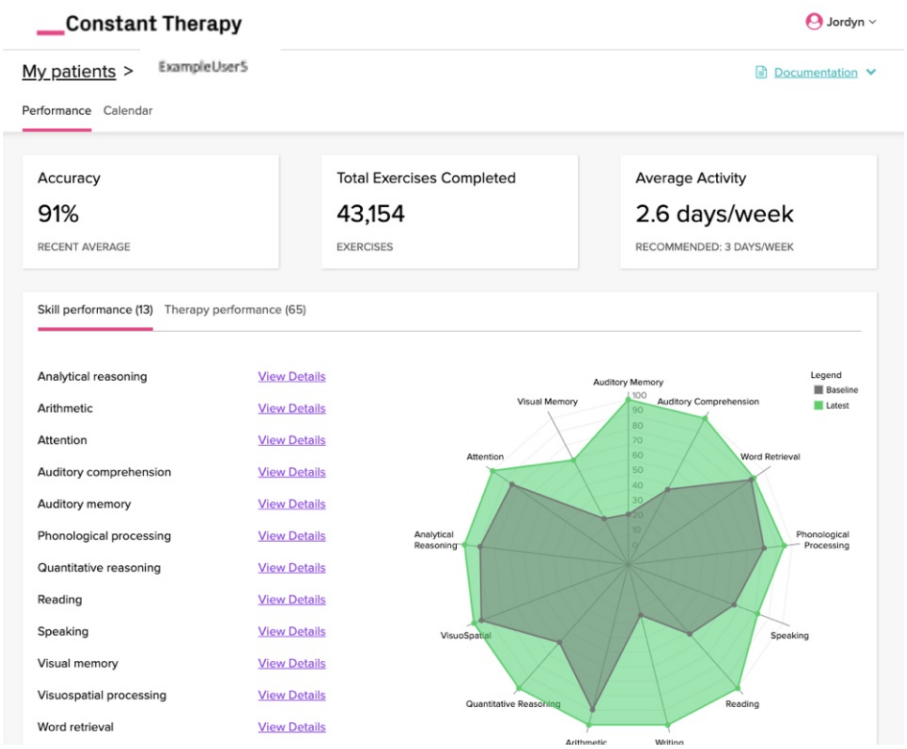


Figure 2 - Screenshot of Constant Therapy Progress Page, Source = https://constanttherapyhealth.com/br

2.2.2 CogniFit

CogniFit utilises many of the tools required for a successful cognitive-based application that many of the other applications use. Unlike the others however, this application makes use of Artificial Intelligence (AI) algorithms in order to aid with user personalisation

based on the results users have achieved from previous cognitive assessments through this application, tailored to the weaknesses of the user.

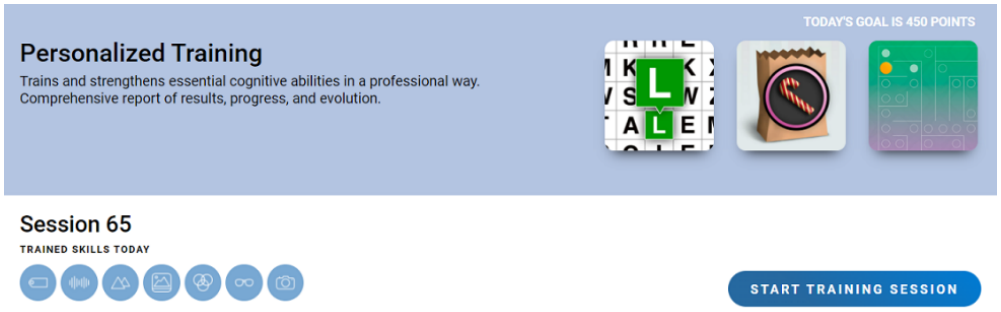


Figure 3 - Screenshot of CogniFit Personalisation Page, Source = https://support.cognifit.com/s/articl

2.3 Gap in the Research

While reviewing research gathered around Dementia and its relations to cognition, it can be seen that Dementia is a vast issue ranging from several different causes and factors that play a role in its development.

However this research study aims to find ways to slow down the cognitive decline as individuals grow older and become more prone to developing this syndrome. By conducting this research and gaining the appropriate data needed, a solution can be developed, providing ways for patients to be able to actively participate in cognitive assessments in an engaging way. This solution will draw inspiration from both the existing solutions (provided in Chapter 2.2) as well as found literature, by not only creating an engaging environment for individuals to improve their cognition but also an interface focusing primarily on learnability as well as efficiency.

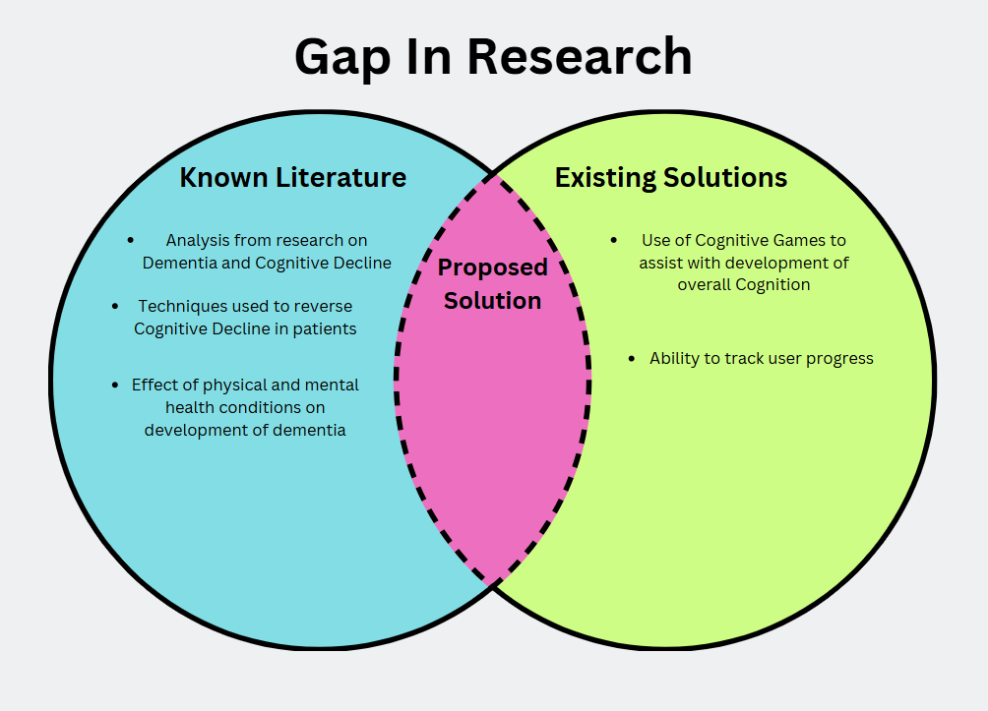


Figure 4 - Gap in Research

The insights gained from Known Literature on this research area takes into account the analysis of dementia, cognitive decline as well as techniques utilised to reverse the decline and its effect on physical and mental health. Existing Solutions aims to gather necessary information from the applications that this solution will take inspiration from. By taking these two insights into account, the Proposed Solution aims to bridge the gap between both insights by integrating necessary data and make a solution that is targeted specifically towards its primary stakeholder.

2.4 Research Questions

A series of research questions will be created in order to better understand the gap in the research. The questions will primarily focus on ways to create a solution that is tailored towards its primary stakeholder as well as one that aims to convey effectiveness in slowing down or even stopping cognitive decline.

- Which cognitive domain declines the quickest in patients suffering from dementia?
- What have existing solutions added into its applications to make its app more engaging?
- How is overall cognition measured in individuals, and how can it be implemented into the solution?
- What physical illnesses or other physical factors play a role in cognitive decline?

Chapter 3: **Methodology, Primary Research and Data Analysis**

The aim of this solution is to measure the efficacy of cognitive games on an individual's overall cognitive function. However, prior to developing this application, primary research for this solution will be carried out in the forms of qualitative and quantitative approaches to gathering necessary data, so that a deeper understanding of the problem this solution can be obtained.

3.1 Methodology

The key primary research methodologies that will be undertaken are surveys and interviews from medical students and doctors who have had sound experiences in dealing with patients suffering from dementia or any form of cognitive impairment. Carrying out these qualitative and quantitative approaches would allow for both an ethical approach to collecting data around this area of research, due to the nature of this research, as well as the ability to collect essential information that will be a significant influence in the formation of the final solution.

3.1.1 Interviews

It can be seen that through several examples of research studies carried out, a primary form of qualitative data collection was interviews with a targeted group of participants. This was carried out so that the researcher could collect detailed, essential and specific data from the participants of the interview. Exploration of various viewpoints and in-depth insights into the specified topic can allow for the collection of extensive and detailed information.

For this research study, interviews will be conducted towards medical students and doctors who have general expertise in the area of this study. Through this approach, an understanding of cognitive challenges in patients can be explored with the additional focus on user experience. This will be carried out so that by identifying the most affected cognitive domain, the final solution can be developed to effectively target this domain. In addition, the added insight of observations of individuals with dementia and cognitive impairment can aid in shaping the solution of this research study.

Moreover, participants of this interview can share their personal points of view on the use of cognitive games in aiding in the improvement of an individual's cognitive decline. An insight into the viewpoints of professionals in this field regarding game-based assessments can help in the development of the application, specifically making it tailored towards the primary stakeholder.

3.1.1 Surveys

In addition to carrying out interviews to gain knowledge about a specific topic, a different approach to collecting necessary information is through quantitative data collection methods such as surveys. This method is viewed as a more efficient and feasible approach to the collection of data in comparison to other methods, as it requires fewer resources and can be carried out online without the researcher or the participant having to physically meet up or travel.

After having carried out interviews with participants, surveys will be conducted with the same participants in order to build on the data collected from the interviews. Primarily, looking into gaining more of an understanding of the cognitive challenges faced as well as a focus on tailoring the solution toward individuals suffering from dementia. Moreover, the data collected from surveys will additionally be used to support the data gathered from interviews so that a general idea may be created in order to best create the solution for the primary stakeholder.

3.2 Primary Research

3.2.1. Overview of Primary Research

After having collected the appropriate data from Medical Students and Doctors around how the application can be best developed to tailor towards the primary stakeholder of the application, dementia patients.

Semi-structured interviews were chosen as a way of interviewing Medical Students and Doctors. Medical Students and Doctors were chosen as individuals to be interviewed as well as collect necessary data in the form of surveys, primarily as it isn't ethically responsible to directly interview dementia patients and put them at risk due to their cognitive impairments. Additionally, Medical Students and Doctors have hands-on practical experience in dealing with dementia patients as well as clinical knowledge on cognitive impairment.

3.2.2. Interview and Survey

The interview questions utilised (**Appendix A**) aimed at trying to understand the most affected cognitive domains in patients suffering from dementia, additional methods of attempting to reverse cognitive decline, their expectations on what the application should achieve for dementia patients, the utilisation of games to improve overall cognition and any ethical considerations that might need to be implemented into the application.

The survey questions (**Appendix B**) were conducted alongside the interview to gain a broader understanding of how the application can be best developed to support the primary stakeholders' needs. The survey collects both quantitative and qualitative data on areas such as the most affected cognitive domains as well as their standpoints on the utilisation of apps and games to aid with cognitive impairment. **Appendix C** displays the survey responses from each of the doctors and medical students, which were used as reference in the early stages of development.

Both the interview and survey were carried out to ensure that the solution was not only appropriate clinically for individuals with dementia but also that it is user-friendly and ethically appropriate.

3.3 Risk Register

It is inevitable that there will be issues and risks that will arise when trying to develop this solution. However, steps must be taken to avoid the chances of these taking place. A risk assessment is displayed, conveying the likelihood of the issue or risk-taking place,

the impact it would have on the development process of the solution as well as necessary action that will need to be taken to mitigate this.

ID	Risk	Likelihood	Impact	Actions
1	Technical Challenges during the Development process	High – This task involves working with complex code, and will definitely involve learning new coding frameworks and encountering several bugs	High – Encountering these issues may lead to certain features not being implemented on the solution, may have a big impact the final grade for this project	Use knowledge gained from frameworks used in previous years to aid in the development of this application, focus initially on the main functionalities that address the aim of this project. Consider utilising lightweight frameworks for this project
2	Participants unavailable for data gathering process	Medium – Doctors and medical students can be extremely busy, and it can be extremely challenging for them to find time to be interviewed	Medium – Limited number of participants could have an impact on the depth of the research insights. Would impact ability to design solution to be tailored towards dementia patients	Find participants from careers which both have experience dealing with dementia patients and are closely related to the careers of the participants, such as nurses or carers.
3	Huge workload from university	High – Trying to complete coursework and revise for exams, while also completing this project will be extremely overwhelming	High – Encountering these issues may lead to certain features not being implemented on the solution, may have a big impact the final grade for this project	Use tools such as Trello to be able to break down and organise tasks and regularly collaborate with supervisor in order to be on schedule.
4	Challenges in usability for dementia patients	Medium – Dementia patients may have issues with using the application	High – Issues with utilising the application can lead to application not being as successful as intended	After having created the application, ask doctors and medical students for their professional opinions on the application for dementia patients.
5	Loss of data	Low – Code for this project will be saved on GitHub, therefore it is unlikely for data to be lost	High – Loss of code could have huge impact on the entire project, resulting in huge impacts to the final grade of this project and could	Alongside code being saved on GitHub, code will also be saved on OneDrive so that it can be more easily accessible through other machines.

			result in delays to the project.	
6	Illness/ Unforeseen Issues	Medium	Medium – Encountering these issues may cause delays with the completion of the project	Create a plan with goals to complete ahead of schedule taking into account the possibility of this scenario.
7	Delays in data collection	Medium – Some doctors and medical students may take longer to complete the surveys with their busy schedules	Medium – Delays in submission of data could also impact delays in the completion of the application	Create strict deadlines for participants to submit their feedback, and also follow up on any participants who haven't submitted surveys after the deadline.
8	Additional improvements for solution	Medium – Upon receiving feedback from surveys and interviews, new features may need to be added that weren't in the initial plan.	High – Going on a different path to the initial plan could possibly result in delays due to further research to employ these additional features.	Focus on adding the main features into the application before adding extra features
9	Feedback not detailed enough	Medium – Some participants may offer vague feedback for surveys or interviews	Medium – Responses that are vague can have a huge effect on creating the solution tailored towards dementia patients	Follow up on participants who have given vague responses to interview or survey questions.
10	Loss of motivation	Medium – Challenges will occur through the development process of the project and even through academia or personal life which may cause loss of motivation for the project.	Medium – Reduced motivation can cause hinder progress	Break down the project into smaller and achievable parts and keep track of progress made and take regular breaks.

Table 2 - Risk Assessment Table

Chapter 4: **Requirements, Design and Architecture**

4.1 Requirement Analysis

By utilising data collected from both the literature review and data gathering methodologies conducted, a list of requirements will be developed to be put into this solution. These requirements will be divided into functional and non-functional. Functional Requirements portray the functionality of the solution, outlining the features and functions it must carry out for the user. Non-Functional Requirements showcase the attributes of the system such as how the system should react when certain conditions take place.

A set of functional and non-functional requirements were created at the initial stages of implementation:

4.1.1 Functional Requirements

ID	Priority	Requirement	Description
F1	Core	Login Feature	Users must be able to login using their email and chosen password.
F2	Core	Register Feature	Users must be able to register for an account with their email to utilise the system.
F3	Core	Progress Track Feature	Users must be able to track their overall progress after having completed assessments on this application, an overview of areas of improvement must be highlighted.
F4	Core	Dashboard	Users must select a cognitive game (Digit Span Test, Self-Ordered Search and Paired Associate Learning) to carry out from the dashboard page
F5	Optional	Reward System	Addition of achievement/trophy system to encourage users to carry on using the application regularly

Table 3 - Initial Stage: Functional Requirements Table

4.1.2 Non-Functional Requirements

ID	Priority	Requirement
NF1	Core	The System must show responsiveness to user action within at least 1 second
NF2	Core	The System must load within 5 seconds
NF3	Core	The System must be run on a web browser
NF4	Core	The System must be able to handle wrong inputs from the user correctly (e.g. invalid login details or incorrect data inputted)
NF5	Core	The Code must adhere to PEP8 code format standards for Python to maintain consistency
NF6	Core	The Code must be written in HTML, CSS, React.js and Python (Flask)
NF7	Core	The data inputted into the system before an update, must remain the same after the update
NF8	Core	The System must put emphasis on strong passwords and secure login standards for users.
NF9	Core	The System must be able to handle multiple users and data being inputted

Table 4 - Initial Stage: Non-Functional Requirements Table

The initial requirements listed above were constructed in the early stages of the development process. In the middle stages of development, new updated functional and non-functional requirements tables were created in **Appendix F**, which portray both the original features implemented in the tables above as well as additional features and requirements that are essential for this system.

4.2 Use Case Diagram

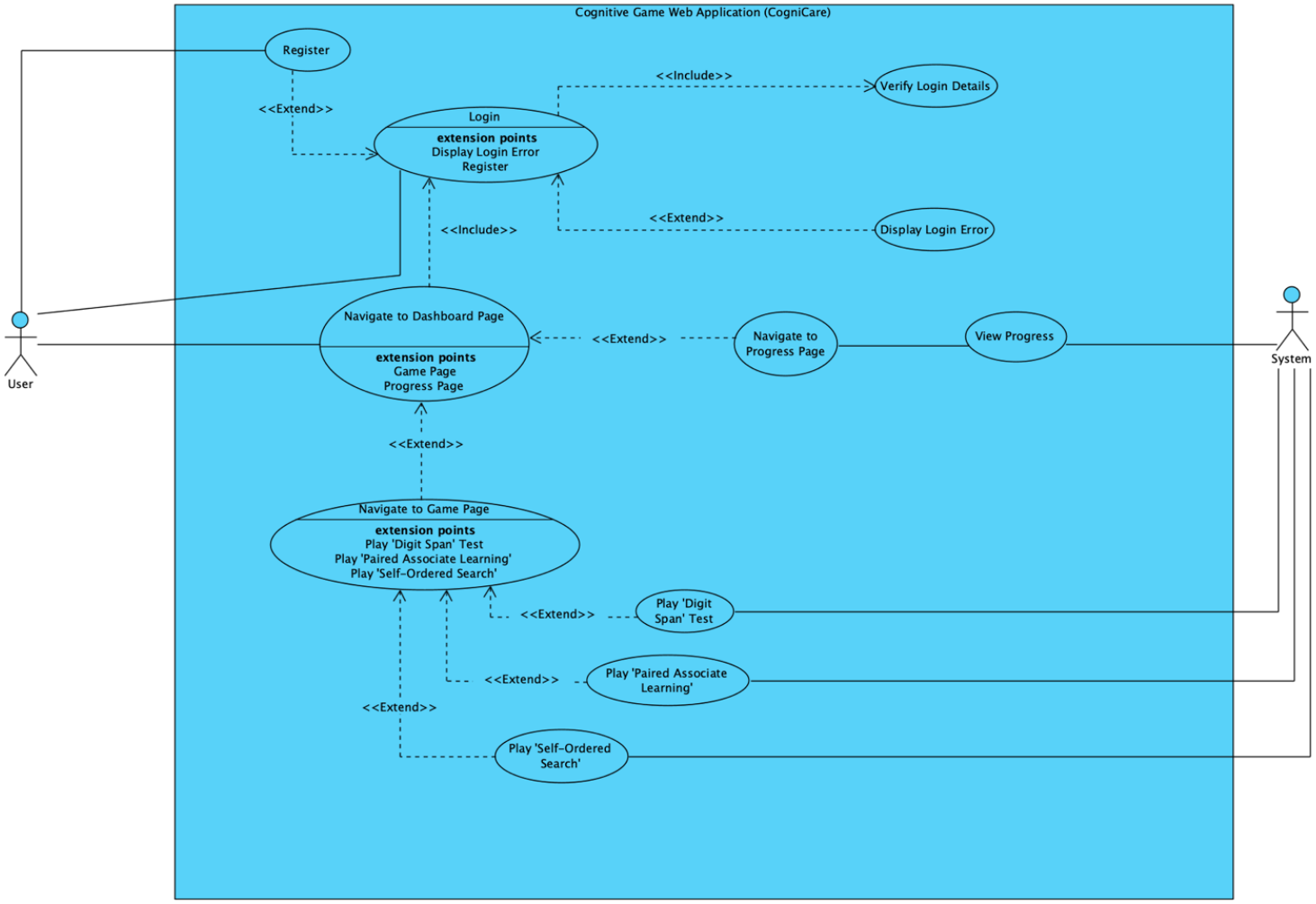


Figure 5 - Use Case Diagram (created using Visual Paradigm)

The use case diagram presented above, conveys the relationships between the use cases of the diagram as well as the roles the actors, the User and the System, play for the web application to function accordingly. Relationships such as the Associate, 'include' and 'Extends' are utilised in this diagram as a way to portray both the next steps the user can take when navigating across the application as well as error handling and information provided by the system so that the user may carry out certain tasks.

As seen in Figure 7, the User actor is shown to have an association relationship with the use cases "Register", "Login" and "Navigate to Dashboard". The user should be able to login to the application and if an incorrect login detail is entered, an appropriate message showing that there was an error in the login details should be displayed to the user. The use case of "Verify Login Details" has an 'include' arrow from the base use case of "Login" as the details entered by the user must be verified before allowing the user to navigate to the application's Dashboard page. When the user is on the Dashboard page, they must have the choice to either navigate to the Progress or Game page, this is showcased using the 'Extend' arrow from the extend use cases: "Navigate Progress Page" and "Navigate to Game Page". From here, the user will be able to either play one of the cognitive games or view their progress.

The System actor must be able to aid the user with the action they wish to carry out. By viewing Figure 7, it can be seen that it has association relationships with the use cases of “View Progress”, “Play ‘Digit Span Test’ “, “Play ‘Paired Associate Learning’ “and “Play ‘Self-Ordered Search’ “. Through the “View Progress” use case, the system should be able to provide the appropriate progress data for the user to view from the Progress Page. Additionally for the other use cases, which are Extended use cases of the base use case “Navigate to Game Page”, the system should be able to show the appropriate game for the selected game.

Chapter 5: **Implementation**

The implementation of this solution will use React.js for the development of the frontend of the application as well as Flask to handle the backend. Through the integration of React.js and Flask, a full-stack web development framework will be created. For this solution, React.js offers the creation of a user-friendly and highly responsive user interface which is extremely beneficial for dementia patients when utilising the application. Combining this with Flask, a lightweight backend Python web framework, allows for the application to have smoother, faster communication with the frontend.

5.1 Project Structure

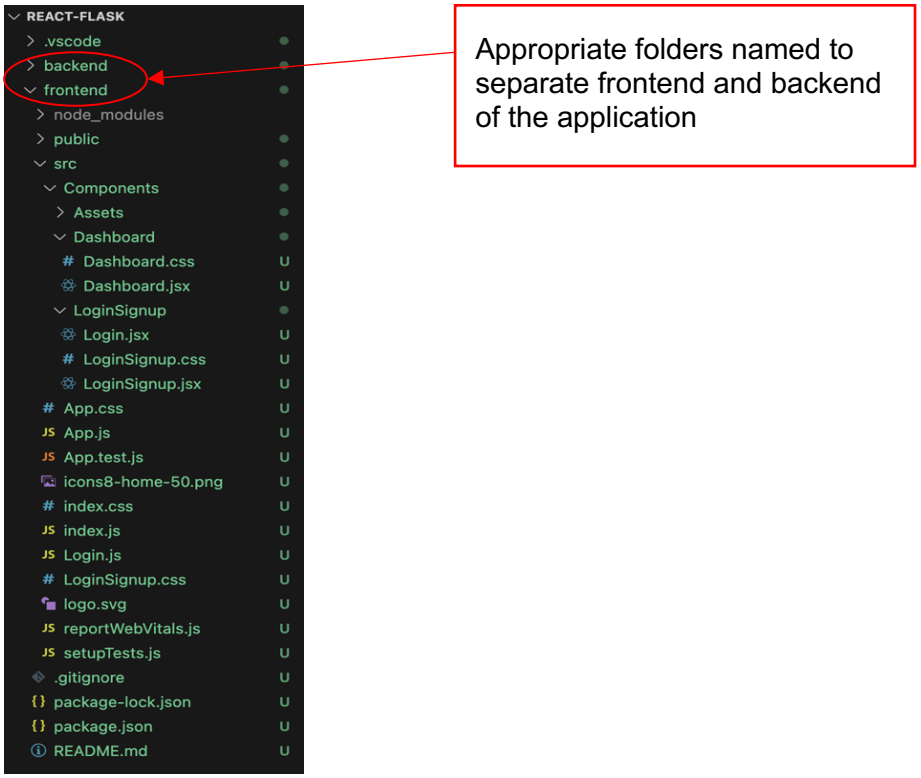


Figure 6 - Code Structure for Project

In order to achieve this coding setup, Visual Studio Code was utilised as a development environment to build this solution. After having created a new folder, Flask was installed initially using the terminal provided by this IDE and moved into the folder labelled 'backend', to allow for better organisation when developing this application when working specifically on the backend for this project using Flask. After having set up the backend for this project, a new folder named 'frontend' is created with the same purpose as the 'backend' folder, to allow for a more streamlined development. In the 'frontend' folder, React.js will be installed through the terminal with the appropriate commands inputted. When the local development server is initiated using the 'npm start' command, after having moved to the folder named 'frontend' using the 'cd' command in the terminal, the application will be displayed in the URL: 'http://localhost:3000/'. This URL is the root URL for the application, which will open the Dashboard page of the web app.

5.1.1 Dashboard Page

```
return (
  <section className="hero-section py-5">
    <h1 className="display-2 fw-bold headertext text-center text-primary mb-4">
      Welcome to CogniCare!
    </h1>
    <h2 className="lead mt-4 subtitle text-center text-muted" id="subtitle">
      Sharpen your memory by playing games and track your progress along the
      way!
    </h2>
    <h3 className="text-center mb-4">Welcome, {login?.username}!</h3>

    <div className="mt-7 text-center">
      <a
        href="games"
        className="btn btn-success btn-lg px-4 py-2 me-3 shadow"
      >
        <GamepadIcon className="w-5 h-5 mr-2" />
        Play Games
      </a>
      <a href="progress" className="btn btn-info btn-lg px-4 py-2 shadow">
        <BarChart3 className="w-5 h-5 mr-2" />
        View Progress
      </a>
    </div>

    <div
      className={`quote-box mx-auto my-4 ${fade ? "fade-in" : "fade-out"}`}
    >
      <p className="quote-text">
        {" "}
        <Lightbulb />
        {quotes[quoteIndex]}
      </p>
    </div>
  </section>
);
```

Figure 7 - React code for Dashboard

After having logged into the application, the user will be directed to the 'Dashboard' page where they are presented with a heading and subheading conveying a welcome message with their chosen name as well as what actions they may take while utilising the application. Since this application is primarily aimed at patients with dementia, who are elderly individuals, the text and buttons presented on this page will be bold, large, clear as well as simplistic, aiding in the reduction of cognitive load on the user.

```
import "bootstrap/dist/css/bootstrap.css";
import "./Dashboard.css";
import { BarChart3, GamepadIcon, Lightbulb } from "lucide-react";
```

Figure 8 - Utilisation of 'lucide-react', CSS and Bootstrap in Dashboard page

To enhance the attractiveness of this page whilst also making it simplistic for the primary stakeholder, Bootstrap alongside CSS and the React library 'lucide-react' is utilised. The CSS was mainly applied for the styling of the text presented on the page, alongside Bootstrap which was implemented to ensure the text presented contrasted the background of the application, ensuring readability by taking into account dementia users who suffer from visual impairment.

Presented on the Dashboard page are two buttons: a ‘Play Games’ button and a ‘View Progress’ button. Both buttons are large with clear text, contrasting colours implemented using Bootstrap and appropriate icons utilising the React library ‘lucide-react’. The displayed buttons allow for simplified navigation across the application with minimal steps taken to traverse from one page to another. This was implemented when taking into account the cognitive challenges that may be experienced by users when using an application, without overwhelming the user with many choices.

```
return (
  <nav className="d-flex justify-content-between align-items-center mb-4">
    <button className="btn btn-lg btn-primary me-3" onClick={handleDashboardClick}>
    <Home className="w-5 h-5 mr-2" />
      Home
    </button>
    <button type="button" className="btn btn-lg btn-danger" onClick={handleSignOutClick}>
    <LogOut className="w-5 h-5 mr-2" />
      Sign out
    </button>
  </nav>
);
```

Figure 9 - Code Snippet of Navigation Bar

In addition, a navigation bar will be displayed with all pages on this application, including the Dashboard page. The navigation bar has a minimalistic design, containing only two buttons: a ‘Sign Out’ button and a ‘Dashboard’ button. Utilising Bootstrap, a contrast in colour was employed, which can be seen in ‘btn-primary’, which displays a blue button and ‘btn-danger’ for a red button. By using ‘lucide-react’ alongside this, a ‘Home’ and ‘LogOut’ icon was implemented in order to assist in understanding the actions each of the buttons carries out.

```
const quotes = [
  // Array of motivational quotes
  "The best way to keep your memories alive is to make new ones.",
  "Memory is like a muscle — the more you use it, the stronger it becomes.",
  "Keep going. You're remembering more than you think.",
  "Every click, every try, every game — it all counts.",
  "Progress, not perfection. Just by being here, you're already improving.",
];
```

Figure 10 - Array of motivational quotes

In the later stages of development, after implementing the core features into the application, an array of motivation quotes was added to the Dashboard page with the purpose of enhancing user experience. To promote regular use of the application, the quotes act as a method of motivating users to play the games consistently to improve overall cognition.

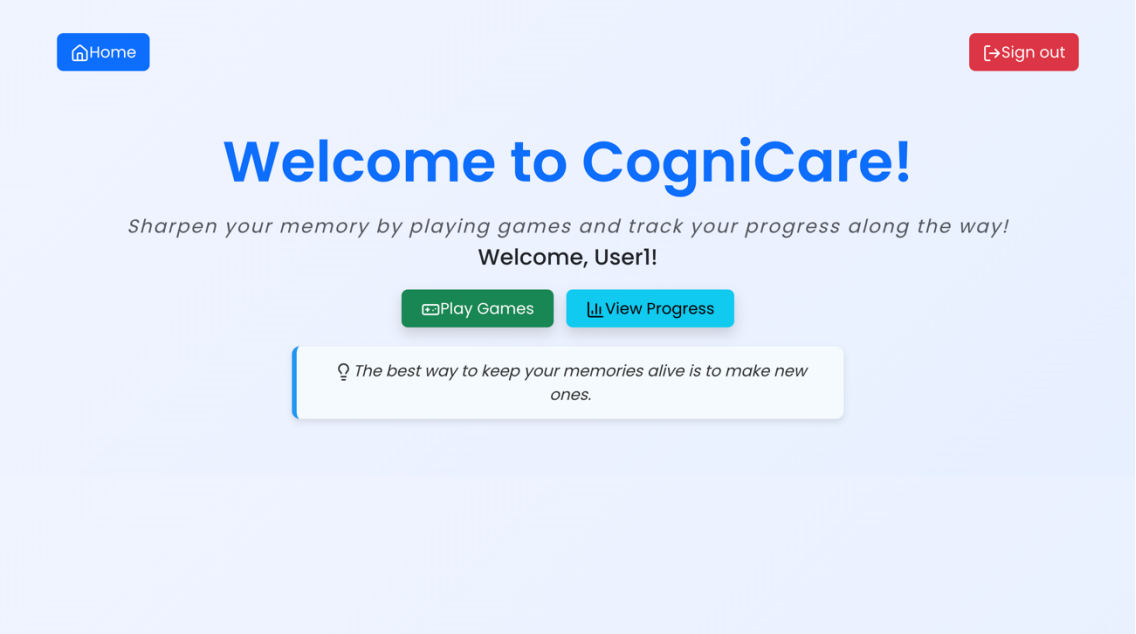


Figure 11 - The Dashboard Page

5.1.2 Progress Page

After having completed the tests or navigating from the 'Dashboard' page, the user can view their progress on each of the games in the 'Progress' Page.

As seen in Figure 14, the Progress page communicates with the backend by fetching user-specific scores and high scores from the backend API, using the 'Bearer' token authentication, which is stored in Session Storage, to be displayed on the frontend to the user.

```
const fetchDigitSpanScore = async () => {
  try {
    const token = JSON.parse(sessionStorage.getItem("login"))?.token;
    const res = await fetch("http://localhost:5000/api/digitspan/score", {
      headers: { Authorization: `Bearer ${token}` },
    });
    const data = await res.json();
    setDigitSpanScore(data.latest_score);
    setDigitSpanHighestScore(data.high_score);
  } catch (err) {
    console.error("Error fetching Digit Span score:", err);
  }
};
```

Figure 12 - Code Snippet of fetching user-specific score and high score for a game

The above code snippet conveys an example of frontend-backend communication from the Progress page to fetch from the server running in the backend: the recent and high scores from the Digit Span Test ('Mathemagician' game). After a user signs into the application, a token is saved in the browser, which identifies who the user is. This aids in attaining user-specific scores. The server replies by giving the scores to the frontend in JSON formatting, which is read by the frontend and stored in memory using the methods 'setDigitSpanScore' and 'setDigitSpanHighestScore'. If an error occurs, an

error message will be printed onto the browser console. This code is the same as fetching user-specific scores for all games in the application.

```
const [digitSpanScore, setDigitSpanScore] = useState(null);
const [digitSpanHighestScore, setDigitSpanHighestScore] = useState(null);
const [pairedAssociateScore, setPairedAssociateScore] = useState(null);
const [pairedDifficulty, setPairedDifficulty] = useState(null);
const [pairedAssociateHighestScore, setPairedAssociateHighestScore] = useState(null);
const [selfOrderedScore, setSelfOrderedScore] = useState(null);
const [selfOrderedHighestScore, setSelfOrderedHighestScore] = useState(null);
```

Figure 13 – 'useState' hooks for storing user-specific scores and difficulty

In order to store user scores and difficulty in the Progress page, useState hooks are used to create variables which initially have a value of 'null'.

```
<div className="progress-section-box">
  {digitSpanScore >= 7 ? (
    <div className="achievement-card bg-success text-white">
      <strong>Achievement Unlocked:</strong>
      <br /> Digit Dynamite – Fantastic memory!
    </div>
  ) : (
    <div className="achievement-card bg-light border border-primary text-dark">
      To become a
      <strong>
        <br /> Digit Dynamite{" "}
      </strong>
      achieve a
      <span className="fw-bold text-danger">
        <br /> score of 7!
      </span>
    </div>
  )}
</div>
```

Figure 14 - Code snippet of trophy system for Digit Span Test ('Mathemagician' game)

Utilising emojis to promote visual attraction, a trophy system was implemented with the intention of enhancing user engagement while using the application, with the aim of motivating users to push themselves to achieve a high score. The above code shows that if a user attains a score of higher than or equal to 7, they will unlock an achievement in the form of a green success card. If they do not attain this achievement, they will be presented with a card with blue borders conveying a message that the user has not achieved a score in the desired range. A similar method was employed for all games on this page.

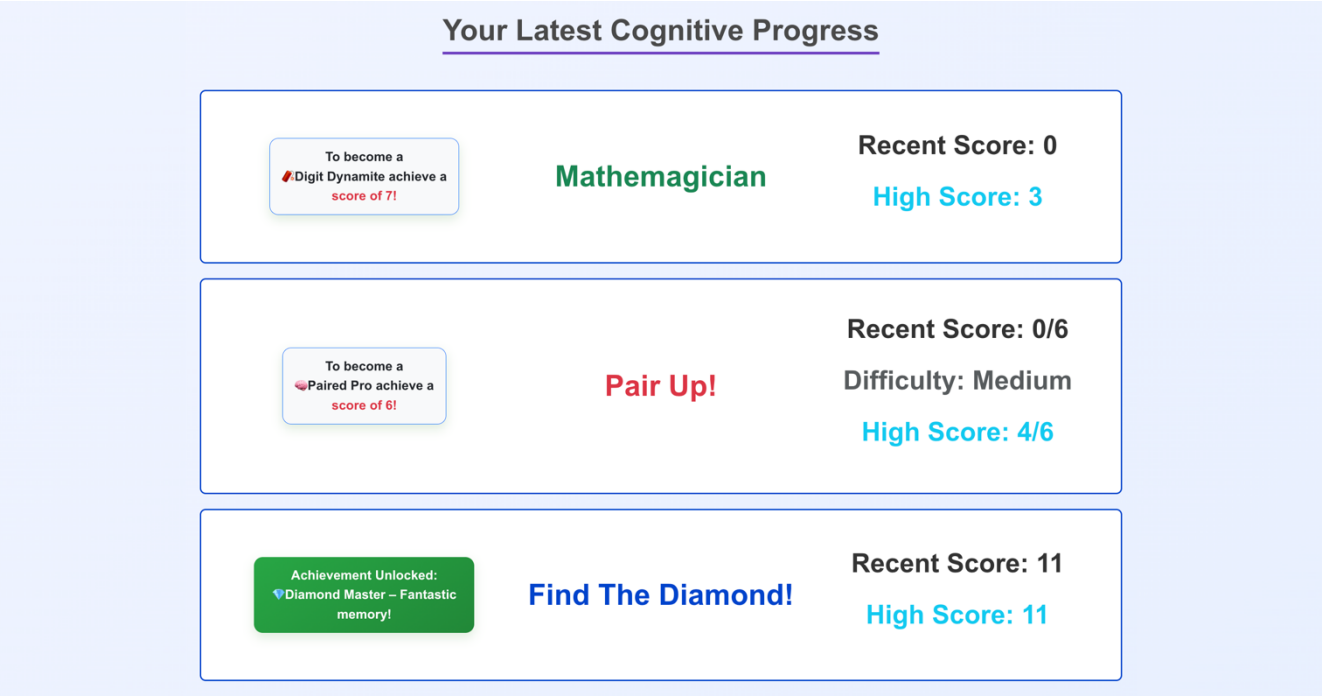


Figure 15 - The Progress Page

5.1.3 Game Page

```
export const GameHub = () => {
  return (
    <section class="hero-section py-2">
      <div className="Title">
        <h1 class="display-2 fw-bold headertext text-center text-danger mb-4">
          Welcome to the Game Page
        </h1>
        <h2 class="lead mt-4 subtitle text-center text-muted" id="subtitle">
          Explore Games to improve your memory
        </h2>
      </div>
      <section class="hero-section py-2">
        <div class="row mt-5">
          <div class="col-md-4">
            <div class="p-4 border rounded bg-success text-light tooltip-container">
              <h4 class="fw-bold">
                {" "}
                <Sparkles /> Mathemagician
              </h4>
              <div class="tooltip-content">
                Challenge your memory by recalling sequences of number that get
                longer in each round! <br />
                <br />
                <b>
                  <u>Watch the video below!</u>
                </b>
                <div className="d-flex justify-content-center">
                  <video
                    src={DigitSpanDemo}
                    poster={DigitSpanPoster}
                    style={{
                      width: "100%",
                      maxWidth: "250px",
                      marginTop: "10px",
                      borderRadius: "8px",
                    }}
                    controls
                  />
                </div>
```

Figure 16 - Code Snippet of the Game Page

After navigating to the ‘Game’ page, like the ‘Dashboard’ page, the user is presented with a heading and subheading styled with a brief welcome message utilising Bootstrap for styling, alongside CSS.

The aim of this page is to enable users to engage with different cognitive-based games in order to enhance specific cognitive abilities. By taking into account the primary stakeholder of this application, individuals with dementia can suffer from various types of cognitive impairments as a result of this cognitive decline can affect individuals differently. In order to resolve this, this application allows users to choose between three different cognitive games, which are aimed at reversing the cognitive decline in the most commonly affected cognitive domains.

Each game: the ‘Digit-Span Test’ (named “Mathemagician”), ‘Paired Associate Learning’ (named “Pair Up!”) and ‘Self-Ordered Search’ (named “Find The Diamond!”), is presented in coloured boxes which contrast each other, with clear text conveying what the game is which is implemented using Bootstrap. Alongside this CSS makes use of the ‘hover’ effect to ensure a dementia-friendly environment, so that when the user hovers their cursor over each coloured box, it expands presenting information about each of the games, a video tutorial conveying how to navigate and play each of the games as well as a ‘Play’ button which directs the user to a page where they can engage in the selected cognitive game. By considering the primary stakeholder, this feature aims to reduce overall confusion and cognitive overload when the user is about to participate in these games as individuals suffering from cognitive impairment are known to have heightened levels of anxiety, specifically when presented with new environments.

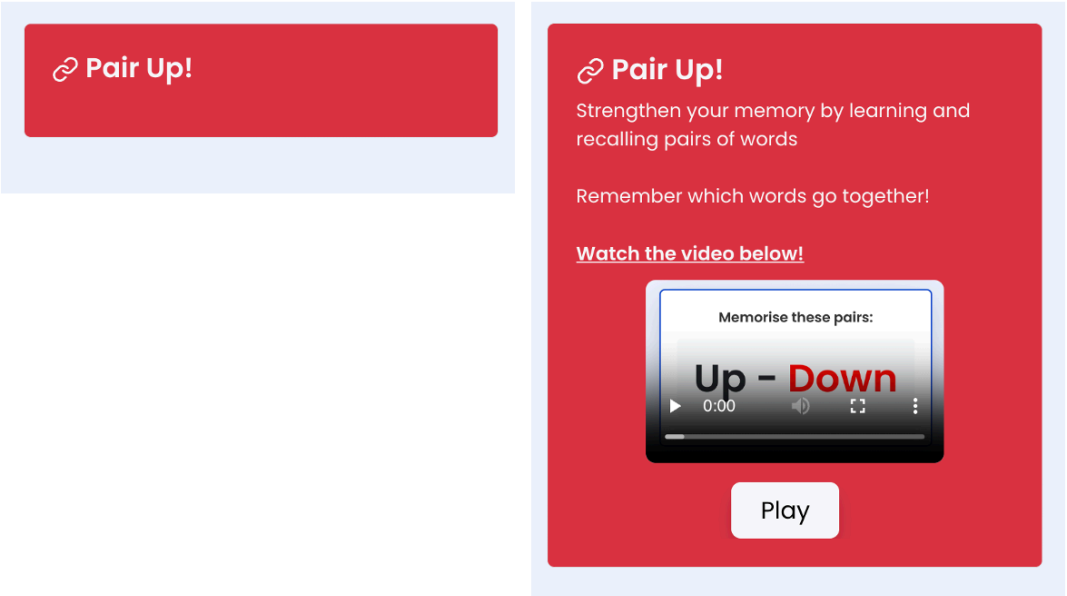


Figure 17 - Before and After: the hover effect is utilised

Overall, the Game Page has a very simplistic yet attractive design, with its simple and overall variation in colour scheme throughout the page. Additionally, there is dementia-friendly navigation throughout the page where users can traverse with ease. The page provides a supportive environment and accessibility to a variety of cognitive-based games targeting different cognitive domains.

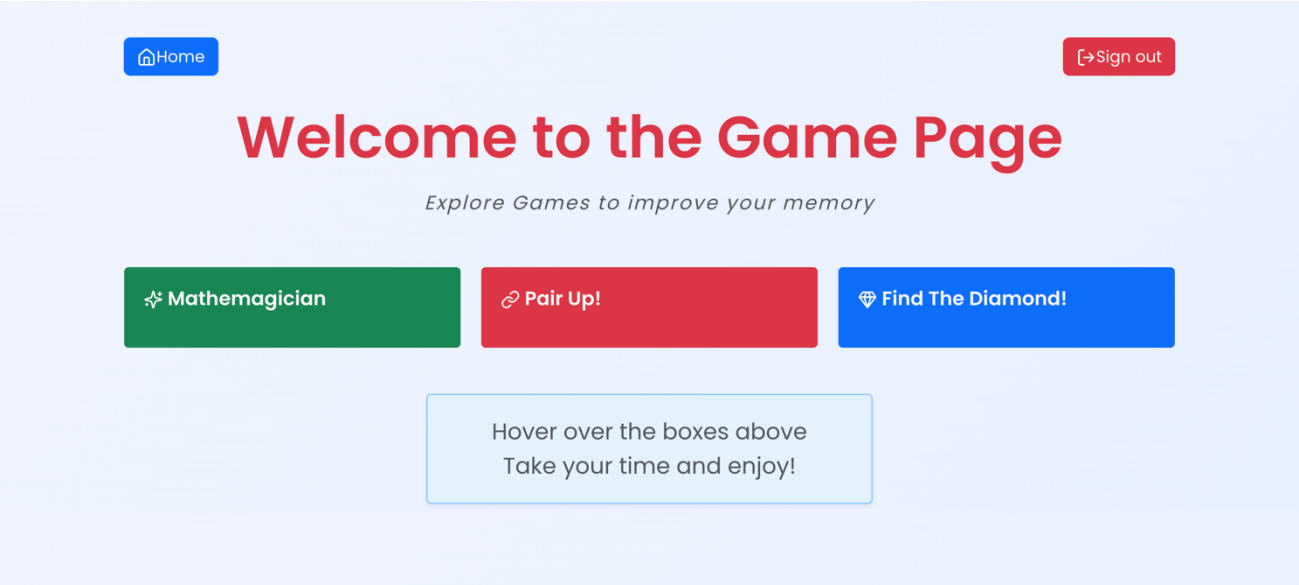


Figure 18 - The Game Page

In the later stages of development, a prompt box was implemented into the Progress Page to further assist users with navigation. The box acts as a form of guidance for the user, providing instructions on interaction with each of the boxes.

5.2 Backend Configuration and Database Models

5.2.1. Backend Configuration

```python
from flask import Flask;
from flask_sqlalchemy import SQLAlchemy;
from flask_cors import CORS;
from datetime import timedelta

app = Flask(__name__)
CORS(app)
app.config['SECRET_KEY'] = "super-secret-key"
app.config['JWT_SECRET_KEY'] = "super-jwt-secret"
app.config['JWT_ACCESS_TOKEN_EXPIRES'] = timedelta(hours=4)


app.config["SQLALCHEMY_DATABASE_URI"] = "sqlite:///mydatabase.db"
app.config["SQLALCHEMY_TRACK_MODIFICATIONS"] = False

db = SQLAlchemy(app)
```

Figure 19 - Backend Configuration code: 'config.py' Code

The code shown above aids in configuring the backend environment of the application. The core of the solution is set up in 'Flask(__name__)' and Cross-Origin Resource Sharing (CORS) is enabled so that frontend-backend communication can take place. Two secret keys are configured within this Python file, where 'SECRET_KEY' is utilised

for session security and 'JWT_SECRET_KEY' is used for handling user authentication by generating JWT tokens. The JWT access token given to the user has an expiration of 4 hours in 'JWT_ACCESS_TOKEN_EXPIRES'. Additionally, SQLAlchemy is configured within the file: the database URI is specified to allow the use of the SQLite database and modification tracking is disabled to improve overall performance and disable events such as event tracking. Both model creation and execution of queries are enabled within the application through 'SQLAlchemy(app)'.

5.2.2. Database Models

```python
class Userinfo(db.Model):
    """
    Model representing user information for login and signup
    """
    id = db.Column(db.Integer, primary_key=True)
    first_name = db.Column(db.String(80), nullable=False)
    email = db.Column(db.String(120), unique=True, nullable=False)
    password_hash = db.Column(db.String(128), nullable=False)

    def set_password(self, password):
        """
        Hashes and stores the password securely
        """
        self.password_hash = generate_password_hash(password).decode("utf-8")

    def check_password(self, password):
        """
        Validates password against stored hash.
        """
        return check_password_hash(self.password_hash, password)

    def to_json(self):
        """
        Returns user data as JSON, excluding password
        """
        return {
            "id": self.id,
            "firstName": self.first_name,
            "email": self.email,
        }
```

Figure 20 - Code for User Model

The class above is used to store user details in the database for functionalities such as logging in and signing up to the application. The four fields listed within the model (id, first_name, email and password_hash) are used to identify each user utilising the system. In addition, two functions are utilised within this class, the 'set_password()' hashes and stores the password for user security, by using the in-built 'generate_password_hash()' function to hash the password before storing it. And the 'check_password()' function verifies the password against the stored hash password when the user logs in by using the built-in 'check_password_hash()' function with the hash password stored in the database.

```python
class UserScores(db.Model):
    """
    Model to track scores for each game per user
    Stores both latest and highest scores, as well as difficulty
    """
    user_id = db.Column(db.Integer, primary_key=True)
    # Digit Span Scores
    digitspan_score = db.Column(db.Integer, nullable=False)
    digitspan_latest = db.Column(db.Integer, nullable=False, default=0)
    # Paired Associate Learning Scores
    pairedassociate_score = db.Column(db.Integer, nullable=False)
    pairedassociate_latest = db.Column(db.Integer, nullable=False, default=0)
    pairedassociate_difficulty = db.Column(db.String(32), nullable=True)
    # Self-Ordered Search Scores
    selfordered_score = db.Column(db.Integer, nullable=False)
    selfordered_latest = db.Column(db.Integer, nullable=False, default=0)

    def to_json(self):
        """
        Returns all user scores in a JSON-serializable dictionary
        """
        return {
            "id": self.user_id,
            "digitSpanHigh": self.digitspan_score,
            "digitSpanLatest": self.digitspan_latest,
            "pairedAssociateHigh": self.pairedassociate_score,
            "pairedAssociateLatest": self.pairedassociate_latest,
            "pairedAssociateDifficulty": self.pairedassociate_difficulty,
            "selfOrderedHigh": self.selfordered_score,
            "selfOrderedLatest": self.selfordered_latest,
        }
```

Figure 21 - Code for User Score model

The above code conveys the User Score model, which stores the latest scores and the highest score of each game as well as the difficulty from the Paired Associate Learning Game (“Pair Up!”).

5.3 Digit Span Test (“Mathemagician” Game)

5.3.1 Implementation

The Digit Span Test is a vital assessment required to improve cognition for dementia patients. Named “Mathemagician” to aid with reducing anxiety so that users do not feel overwhelmed by clinical terminology, this game presents a series of numbers to the user, and the user will have to recall this sequence. Through the progression of this assessment, the sequence will get longer and longer, putting the user’s short-term memory and attention to the test. The maximum number of iterations a user was able to achieve in this assessment will determine their overall digit span score.

Before developing the game itself, different phases of the game had to be implemented in order to ensure a clear structure so that any confusion from the user when taking part

in this game can be mitigated. For this game, four game states had to be introduced, which were listed as: a ‘Start’, ‘Input’, ‘Showing’ and ‘Result’ page.

5.3.1.1. Game States

In order to implement game states, the ‘useState’ hook had to be utilised to manage different phases of the game.

```
const [gameState, setGameState] = useState('start')
```

Figure 22 - ‘useState’ use to set default game state, which is ‘start’

The initial game state that the user should see is the start page for the game. The ‘gameState’ variable holds the value of the current game state that is presented to the user. Initially it is set to ‘start’ and will change accordingly based on the actions executed by the user. The second parameter consists of the function ‘setGameState’, which is utilised alongside the variable ‘gameState’ by updating its value, so that whenever the function is called it updates the value of ‘gameState’.

```
// Start Screen of Digit Span Game
if (gameState === 'start') {
    return (
      <div className="d-flex mt-5 align-items-center justify-content-center vh-100 bg-gradient-to-br from-blue-100 to-green-100">
        <div className="card text-center p-5 shadow-lg" style={{ maxWidth: '600px' }}>
          <div className="d-flex justify-content-center mb-4">
            <Sparkles size={64} className="text-success" />
          </div>
          <h2 className="display-4 mb-4 text-success fw-bold">Mathemagician</h2>
          <i><h2 className="display-4 mb-4 text-success">Can you remember the magic numbers?</h2> </i>
          <p className="fs-4 mb-4">
          🧠 Welcome to Mathemagician! 🔢< br/>
            <br/><ol>
              <li> A short line of numbers will appear </li> <br />
              <li>Watch carefully and remember the <u>sequence of numbers</u> and type them back in order. 🤔</li> <br />
              <li> The more you remember the stronger you become!! 💪</li>
            </ol>
          </p>
          <p className="fs-5 mb-5 text-muted">
            Each correct answer increases the sequence length!
          </p>
          <button onClick={startGame} className="btn btn-primary btn-lg">
            Start Game
          </button>
        </div>
      </div>
    );
  }
```

Figure 23 - React Code showing the Start Page for the Digit Span Test

When the game state is set to ‘start’, simple instructions, alongside emojis to enhance a calming, fun environment for the user, inside a box, convey what the user should expect when attempting to complete the game. After having read the instructions for this game, the user can click on the button titled ‘Start Game’. At this point, the game state will change to a different state, and the user will be directed to a different page of this game.

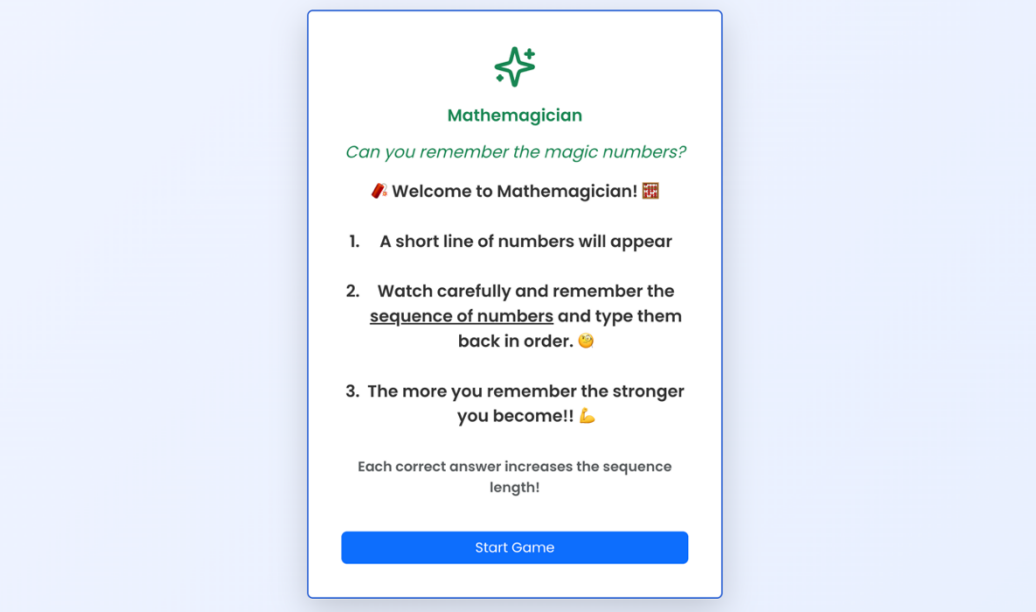


Figure 24 - The ‘Start’ Game State

```
{/* Showing Screen for Number Sequence */}
{gameState === 'showing' && (
    <div className='text-center'>
        <p className="fs-5 mb-3"> Remember this sequence: </p>
            <div className="display-4 fw-bold bg-light p-4 rounded mb-3">
                {numSequence.join(' ')}
            </div>
        <p className="text-muted">
            Time remaining: {userLevel} seconds
        </p>
    </div>
)}
```

Figure 25 - React Code for the ‘Showing’ Game State

After having clicked the ‘Start Game’ button, the user is navigated to the ‘showing’ game page, where they are presented with a number sequence which they will have to remember within a certain period of time, depending on the level they are on. Alongside the change of game state, a random number sequence has to be generated, and a timer has to be inputted relative to the level the user is currently on.

```
// Start Digit Span Game
const startGame = () => {
    setScore(0)
    setLevel(3)
    createSequence()
    setGameState('showing')
    setInput('')
}
```

Figure 26 - Function called when user starts the game

After the user clicks on the 'Start Game' button, when the game state is set to 'start', a series of functions are called. When the Digit Span game ("Mathemagician") starts: the initial score of the game is set to 0, the user will be a presented with a sequence of three digits in 'setLevel(3)', where a number sequence is created with an initial length of three, the game state will change from 'start' to 'showing' and finally a user input feature will be introduced for the user to input the numerical sequence that they must recall, this is set to initially have an empty string value.

5.3.1.2. Number Sequence

```
// Create a random sequence of numbers
const createSequence = () => {
    const Sequence = Array.from(
        {length: userLevel},
        () => Math.floor(Math.random() * 10)
    );
    setNumSequence(Sequence);
}
```

Figure 27 - React Code for generating a number sequence

```
const [userLevel, setLevel] = useState(3)
```

Figure 28 – 'useState' baseline setting for the number sequence

When the game state is presented as 'showing', a random sequence of numbers must be generated for the user to recall. By utilising the baseline setting implemented at the start of this file, the initial value stored in the variable 'userLevel' can be passed into the 'createSequence()' function. The number sequence will have an initial length of three, and digits from 0 to 9 are implemented into the sequence. The code uses 'Array.' to create an Array instance from an array-like object for the digits generated; each of these will be an element within the array.

```
// Create a random sequence of numbers
const createSequence = () => {
    const Sequence = Array.from(
        {length: userLevel},
        () => Math.floor(Math.random() * 10)
    );
    setNumSequence(Sequence);
    console.log(Sequence)
}
```

Figure 29 - React code for creating a random sequence of numbers

```
const [numSequence, setNumSequence] = useState([])
```

Figure 30 – 'useState' hook for 'numSequence' random number sequence

A random sequence of numbers is created relative to the level the user is currently on, random numbers from 0 to 9 are added to an array and are stored in the state variable 'numSequence'.

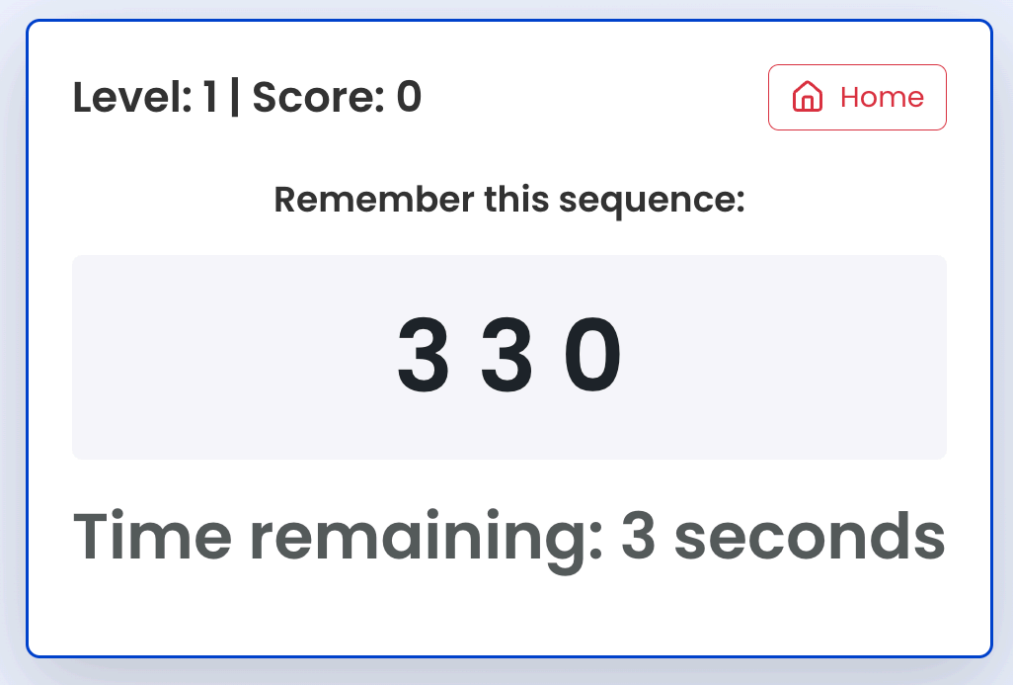


Figure 31 - The 'Showing' Game State

5.3.1.3. User Input

```
{/* UserInput */}
{gameState === 'input' && (
    <div className='text-center'>
        <input
        type='number'
        value={userInput}
        onChange={(e) => setInput(e.target.value)}
        placeholder='Enter Number Sequence'
        className='form-control form-control-lg mb-3'
        autoFocus
        />
        <button onClick={handleUserAnswer} className='btn btn-success btn-lg w-100'>
            Submit
        </button>
    </div>
)}
```

Figure 32 - React Code for 'Input' game state

After the game state changes from 'showing' to 'input', the user is presented with an input box where they are instructed to recall and enter the number sequence that was presented when the game state was set to 'showing'. After having clicked on the 'Submit' button, their response will be stored in 'setInput()' and handled in the 'handleUserAnswer' function.

```
const [userInput, setInput] = useState('')
```

Figure 33 - Baseline setting for user input

```
const handleUserAnswer = () => {
    const isCorrect = numSequence.join('') === userInput;

    // If user input is correct
    if (isCorrect){
        setScore(userScore + 1)
        setLevel(userLevel + 1)
        createSequence()
        setGameState('showing')
        setInput('')
    }
    else{
        // Navigate to result page & show results to user
        sendScoreToBackend(userScore); // Send recent score to the backend

        if (userScore > highScore) {
          setHighScore(userScore); // Update local high score if new score is higher
        }

        setGameState('result')
    }
}
```

Figure 34 - React Code for handling user input

After having input the recalled number sequence into the input box, the input value will be stored and handled using the 'setInput()' function, which updates the value stored in the variable 'userInput', as seen in Figure 24. Since this variable is stored globally, it can be utilised and accessed in the 'handleUserAnswer' function, where the input of the user is compared to the number sequence presented to the user in the 'showing' state, and this will be stored as a Boolean value in the 'isCorrect' variable. The '.join('')' method is used to merge all the elements in the 'numSequence' array into a string value. If the user input value is seen to be the same as the number sequence, both the score and level will increment, a new number sequence will be created with a longer length, the game state will now change from 'input' to 'showing' and the user input will be set to empty. However, if the user input doesn't match the number sequence, their score is passed to the function 'sendScoreToBackend()', which sends the score to the backend, and if their current score is higher than their highest score, this score will be updated as their new highest score using 'setHighScore'. From here, the game state will change to 'result' and the user will be directed to the 'Results' page.

```python
def update_digitspan_scores(user, new_score):
    """
    Updates the Digit Span score for a user. If higher than previous, replaces high score.
    """
    user_score = UserScores.query.filter_by(user_id=user.id).first()

    if user_score:
        user_score.digitspan_latest = new_score
        if new_score > user_score.digitspan_score:
            user_score.digitspan_score = new_score
    else:
        # First time playing - initialize all scores
        user_score = UserScores(
            user_id=user.id,
            digitspan_score=new_score,
            digitspan_latest=new_score,
            pairedassociate_score=0,
            pairedassociate_latest=0,
            selfordered_score=0,
            selfordered_latest=0
        )
        db.session.add(user_score)

    db.session.commit()
```

Figure 35 Flask code for updating user scores

The 'update_digitspan_scores()' function not only updates the user's latest score, but also updates their highest score as well, if their new score is higher. By querying the UserScore table in 'models.py', the function checks if the user's highest and latest score already exists within the table using the user's ID, their scores are updated, and a check is made as well to see if their latest score is higher than their highest score. However, if their score record doesn't exist within the UserScore table, a new user score record is created with other games having their default scores and Digit Span being updated with new scores.

5.3.1.4. Timer

```
// Use of 'useEffect' hook for timer functionality
useEffect(() => {
    let timer;
    if (gameState === 'showing'){
        timer = setTimeout(() => {
            setGameState('input')
        }, 1000 * userLevel)
    }
    return () => clearTimeout(timer)
}, [numSequence, gameState, userLevel])
```

Figure 36 - React Code for timer functionality

In addition to the other features added to create the Digit Span Test, a timer functionality must be implemented to change game states from ‘showing’ to ‘input’, therefore allowing the user to be able to recall the number sequence.

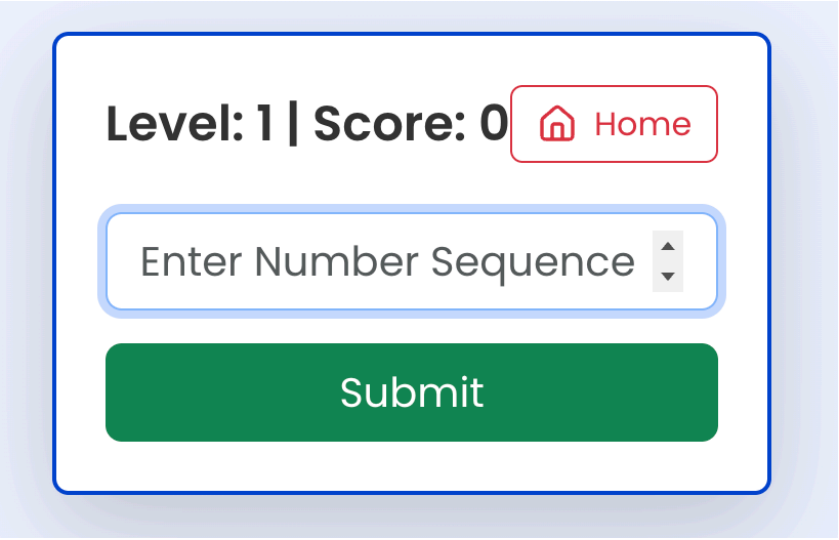


Figure 37 - The ‘Input’ Game State

5.3.1.3. Fetching Highest Score

```
// Fetch user's highest score from backend
useEffect(() => {
  const fetchScore = async () => {
    try {
      const token = JSON.parse(sessionStorage.getItem("login"))?.token;
      const res = await fetch("http://localhost:5000/api/digitspan/score", {
        headers: { Authorization: `Bearer ${token}` },
      });
      const data = await res.json();
      setHighScore(data.high_score || 0);
    } catch (err) {
      console.error("Error fetching high score:", err);
    }
  };
  fetchScore();
}, []);
```

Figure 38 - 'fetchScore' function for fetching Highest score from backend

To enhance user experience by allowing users to track their best performance on a cognitive game, the frontend fetches and displays the highest score to the user on the results page. The function 'fetchScore()' is run inside the React hook 'useEffect()', ensuring that this function is run as soon as the React component is mounted. The fetch request conveyed in the figure above shows the variable 'token' which accesses the JWT token stored in session storage under the key "login", allowing to retrieve user-specific scores, which are parsed and then used in the Authorisation header in the code above. After getting a response from the backend, the frontend converts the response to JSON and the 'high_score' field, from the backend, which stores the user's highest score, is extracted and stored in 'setHighScore'.

```
@app.route('/api/digitspan/score', methods=['GET'])
@jwt_required()
def get_digitspan_score():
    """
    API route to get the latest and high Digit Span scores.
    """
    current_user_email = get_jwt_identity()
    user = Userinfo.query.filter_by(email=current_user_email).first()
    user_score = UserScores.query.filter_by(user_id=user.id).first()

    if not user_score:
        return jsonify({'latest_score': 0, 'high_score': 0})

    return jsonify({
        'latest_score': user_score.digitspan_latest,
        'high_score': user_score.digitspan_score
    })
```

Figure 39 - Flask code for getting highest score

The code above conveys the backend route for retrieving the latest and highest scores for the Digit Span Test (“Mathemagician”) for each user. The route is protected using ‘@jwt_required’, which allows only authenticated users to have access. The user’s information is verified through ‘get_jwt_identity()’, then their email is used to query the ‘Userinfo’ model to get the user’s ID. By using the retrieved ID, a score entry is looked for in the ‘UserScores’ table. From here, a JSON response containing two fields for the latest and highest score is returned.

5.3.1.4. Results Page

```
{/* Results Screen */}
{gameState === 'result' && (
    <div className="text-center" >
    <h2 className="fs-2 fw-bold text-danger mb-4">Game Over!</h2>
    <div className="bg-light p-4 rounded mb-4">
      <p className="fs-4 mb-2" >Final Score: {userScore}</p>
      <p className="fs-4 mb-1">Max Level: {userLevel - 2}</p>
      <p className="fs-4 mb-1 text-success">High Score: {highScore}</p>
    </div>
    <button onClick={startGame} className="btn btn-primary btn-lg d-flex align-items-center gap-2 mx-auto">
    <RotateCcw size={20} />
     Play Again
    </button>
  </div>
```

Figure 40 - React code for ‘Result’ game state

By using Figure 26, the user’s input is compared to the number sequence (which is stored in Array form), and if the user’s input is found to be different from the number sequence, the game state will change to ‘result’, navigating the user to the results page and showing them their scores. From here, the user will have their final score, highest score as well as the level they reached presented to them and in addition to this, they will be given the options to either navigate to the ‘start’ game state or play the game again.

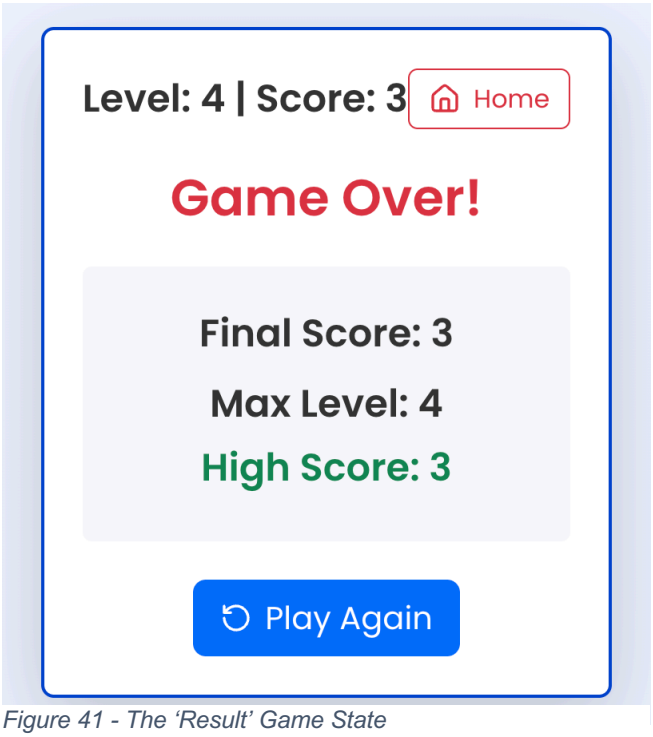


Figure 41 - The ‘Result’ Game State

5.3.1.5. Sending user scores to the backend

```
const sendScoreToBackend = async () => {
  const token = JSON.parse(sessionStorage.getItem("login"))?.token;

  try {
    const res = await fetch("http://localhost:5000/api/digitspan/score", {
      method: "POST",
      headers: {
        "Content-Type": "application/json",
        Authorization: `Bearer ${token}`,
      },
      body: JSON.stringify({ score: userScore }),
    });

    const data = await res.json();
    console.log("Digit Span score submitted:", data);
  } catch (error) {
    console.error("Error submitting score:", error);
  }
};
```

Figure 42 – 'sendScoreToBackend()' function: sends user scores to the backend

After the user is directed to the Results page of the Digit Span Test game ("Mathemagician"), the frontend sends the user's score to the backend of the application. The user-specific JWT token is retrieved from the browser, which is stored in the key "login", for user authentication. A POST request is made to the appropriate backend endpoint. Once received by the backend, a response is given back to the frontend through console logs.

```
@app.route('/api/digitspan/score', methods=['POST'])
@jwt_required()
def submit_digitspan_score():
    """
    API route to submit a Digit Span score (POST).
    """
    current_user_email = get_jwt_identity()
    user = Userinfo.query.filter_by(email=current_user_email).first()
    data = request.get_json()
    score = data.get('score')

    update_digitspan_scores(user, score)
    return jsonify({'message': 'Digit Span score saved'}), 200
```

Figure 43 - Flask code for submitting Digit Span score

The function above handles the submission of the Digit Span score using a POST request. As seen earlier, this function is also protected using '@jwt_required', ensuring

only authenticated users have access. The user's information is accessed using 'get_jwt_identity()' which is then used to query the UserInfo table in 'models.py' to get the user's record. The score is then extracted using 'request.get_json()' and then passed to the 'update_digitspan_score()' function which updates the latest score for the Digit Span Test ("Mathemagician" game) in the database.

The backend logic for submitting, updating and fetching scores as well as the frontend implementation for submitting and fetching scores, are the same for all the cognitive games in the application.

5.4. Paired Associate Learning ("Pair Up!" Game)

5.4.1. Implementation

The Paired Associate Learning is an essential assessment in improving the cognitive domain of 'Learning and Memory' in an individual. Named "Pair Up!" to avoid the user being overwhelmed by clinical terminology, this game involves the user being presented with pairs of items: a stimulus and a response, which must be recalled after a slideshow of all pairs is shown. For example, several stimulus-response pairs will be presented to the user and one of the pairs presented is 'Sun-Hot', from this the user will be shown the stimulus 'Sun' to which they will have to recall the appropriate response to this, which is 'Hot' by remembering the stimulus-response pair presented initially.

Similar to how the 'Digit Span Test' ("Mathemagician" Game) was developed, this game will involve different game states. Ensuring consistency throughout all games on this web application, in turn attempting to minimise confusion with users.

5.4.1.1. Stimulus-Response Pairs

For the Paired Associate Learning game, it is vital to implement several stimulus-response pairs, as this assessment revolves around the user being presented with a stimulus and a response and, in turn recalling the response, this is done to improve overall memory in an individual.

```
const wordPairs = { // Stimulus-Response Pairs organised in difficulties
    easy: [
        {stimulus: 'Hot', response: 'Cold'},
        {stimulus: 'Day', response: 'Night'},
        {stimulus: 'Dog', response: 'Cat'},
        {stimulus: 'Up', response: 'Down'},
        {stimulus: 'Cute', response: 'Ugly'},
        {stimulus: 'Tea', response: 'Coffee'}
    ],
    medium: [
        {stimulus: 'Sea', response: 'Wave'},
        {stimulus: 'Hands', response: 'Feet'},
        {stimulus: 'Raw', response: 'Cooked'},
        {stimulus: 'Water', response: 'Hydrated'},
        {stimulus: 'Tree', response: 'Flower'},
        {stimulus: 'Salt', response: 'Pepper'}
    ],
    hard: [
        {stimulus: 'Algorithm', response: 'Procedure'},
        {stimulus: 'Flower', response: 'Bloom'},
        {stimulus: 'Sealion', response: 'Penguin'},
        {stimulus: 'Desktop', response: 'Laptop'},
        {stimulus: 'Biology', response: 'Catalyst'},
        {stimulus: 'Grid', response: 'Circuit'}
    ]
```

Figure 44 - React Code for Stimulus-Response Pairs

Several stimulus-response pairs were implemented into the code with it being divided into three groups, which the users can select in the ‘Start’ page of this game: ‘easy’, ‘medium’ and ‘hard’. By taking into account that overall cognitive performance varies from user to user, this will allow for there to be a platform where tasks can be appropriately challenging for each user, and there can be a more precise evaluation of each user’s memory function in the Cognitive Domain of ‘Learning and Memory’.

5.4.1.2. Game States

In order to ensure consistency throughout all the games within this application, game states, like from the Digit Span Test, will be implemented into the Paired Associate Learning: ‘start’, ‘slideshow’, ‘quiz’ and ‘result’ game states.

```
const PairedAssociateLearning = () => {
    const [gameState, setGameState] = useState('start')
```

Figure 45 - React Code for initial game state

The initial game state is set to ‘start’, since this is the game page that will be presented to the user after having navigated to the ‘Paired Associate Learning’ (“Pair Up!”) Page from the ‘Game Page’.

```
{gameState === 'start' && (
    <div className="text-center">
        <Link size={64} className="text-danger mb-4" />
        <h2 className="display-5 text-danger fw-bold">Pair Up!</h2>
        <i><h2 className="display-5 text-danger">Can you remember which words go together?</h2></i>
        <p className='fs-4'> 🧠 Welcome to Pair Up! 🧠 </p> <br />
        <p className='fs-4'>
            1. You'll see word pairs – one black, one red ⬛ 🟥 <br/>
            <br />2. Try to remember which red word goes with each black word! 🤔 <br/>
            <br/>3. When the black word appears <u>type in the matching red word from memory! </u><br/>
            <br/>4. Match them right and show off your memory magic! ✨ <br />

        </p>
        <hr />
        <p className="display-1 text-muted">Pick a difficulty level to begin:</p>
        <div className="btn-group">
            <button onClick={() => startGame('easy')} className="btn btn-success btn-lg">Easy</button>
            <button onClick={() => startGame('medium')} className="btn btn-warning btn-lg">Medium</button>
            <button onClick={() => startGame('hard')} className="btn btn-danger btn-lg">Hard</button>
        </div>
    </div>
)}
```

Figure 46 - React Code for ‘Start’ game state

The user will be presented with a start page like the format of the Digit Span Test (“Mathemagician” game) with the use of emojis for visual attraction, however a different set of instructions are presented to them with a set of three difficulties for the user to select from. In order to proceed from the ‘Start’ state, the user will have to select one of the three difficulties where they will be navigated to a different page.

```
const startGame = (chosenDifficulty = 'easy') => {
    setDifficulty(chosenDifficulty);
    const chosenPairs = wordPairs[chosenDifficulty];
    setPairs(chosenPairs);
    setScore(0);
    setSlideshowIndex(0);
    setInput('');
    setQuizIndex(0);
    setQuizOrder([]);
    setGameState('slideshow');
}
```

Figure 47 - React Code for ‘startGame’ function

The default parameter for the chosen difficulty is set to ‘easy’, if no choice is made by the user, however if the user decides to select a specific difficulty, it will be passed to the ‘setDifficulty’ function which will handle this input. Additionally, the user will be traversed to a page where they will be presented with a slideshow of stimulus-response pairs they will have to recall later on in the cognitive game.

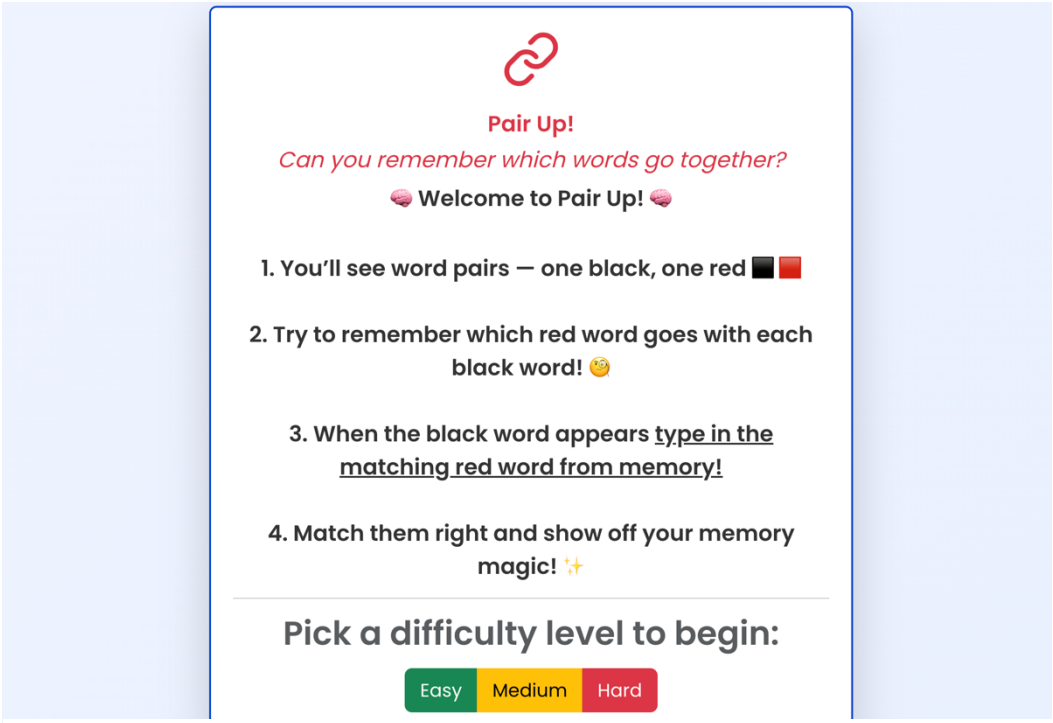


Figure 48 - The ‘Start’ Game State: Paired Associate Learning

Similar to the presentation of the Digit Span Test (“Mathemagician” game), the user is presented with the name of the cognitive game as well as set instructions on what they would expect. Moreover, the addition of a preferred difficulty section allows for the game to be more tailored toward the user’s cognitive level.

```
{gameState === 'slideshow' && slideshowIndex < pairs.length && (
    <div className="text-center">
        <p className="fs-5 mb-3">Memorise these pairs:</p>
        <div className="display-4 fw-bold bg-light p-4 rounded mb-3">
            {pairs[slideshowIndex].stimulus} - <span style={{ color: 'red' }}>{pairs[slideshowIndex].response}</span>
        </div>
    </div>
)}
```

Figure 49 - React Code for ‘Slideshow’ Game State: Paired Associate Learning

```
// useEffect is utilised to aid with 'slideshow' game state
useEffect(() => {
    if (gameState === 'slideshow') {
        if (slideshowIndex < pairs.length) { // If the current index in the slideshow is less than no. pairs
            const timer = setTimeout(() => {
                setSlideshowIndex(prev => prev + 1);
            }, 2000); // Timer is used so that after 2 seconds, next pair is shown
            return () => clearTimeout(timer);
        } else {
            setQuizOrder(generateQuizOrder(pairs)); // If all pairs are shown, start 'quiz'
            setGameState('quiz');
        }
    }
}, [gameState, slideshowIndex, pairs])
```

Figure 50 - Use of timer functionality in ‘slideshow’ game state

The user is navigated to the ‘slideshow’ state, where they are presented with a slideshow of stimulus-response pairs, after selecting their preferred difficulty in the ‘start’ game state. The code validates if the current game state the user is in is ‘slideshow’ as well as ensuring that there are remaining pairs to present in the slideshow, this is carried out through ‘slideshowIndex' which keeps track of the current pair that is displayed and ‘pairs.length’ which returns the total number of pairs, ensuring that the ‘slideshowIndex’ is less than the ‘pairs.length’. If the condition is true, stimulus-response pairs will be presented to the user with the stimulus being black and the response being red.

Utilising the code presented in Figure 52, with the condition that the current index in the slideshow is less than the total number of pairs, a timer is set so that each pair is shown for two seconds, after which the next pair will be shown. After all pairs are displayed to the user, the game state will change to ‘quiz’ where the user will have to recall the response to the appropriate stimulus.

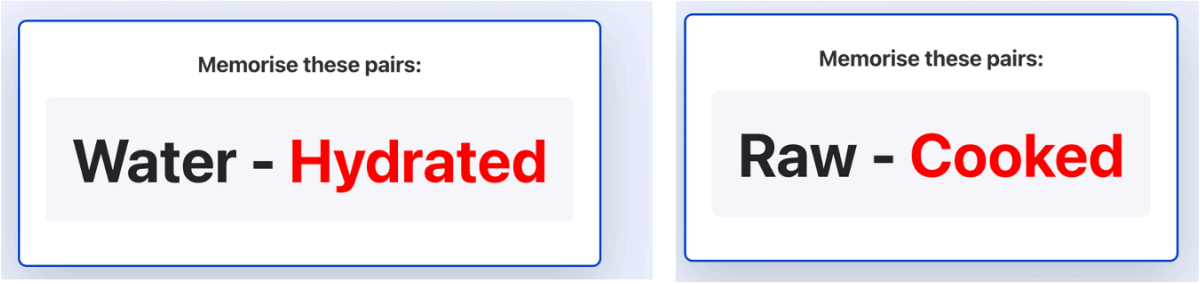


Figure 51 - The 'Slideshow' Game State: Paired Associate Learning

The same layout is utilised from the 'showing' game state of the Digit Span Test ("Mathemagician" game) for this game. However, for this scenario, after being presented with the pairs in a slideshow the user must recall the response pair, the text presented in red.

```
{gameState === 'quiz' && quizOrder.length > 0 && (
    <div>
        <p className='fs-5 mb-3'>What is the correct pair for:</p>
        <div className="display-4 fw-bold bg-light p-4 rounded mb-3 text-primary">{pairs[quizOrder[quizIndex]].stimulus}</div>
        <input
            type='text'
            value={userInput}
            onChange={(e) => setInput(e.target.value)}
            placeholder='Enter the associated word'
            className='form-control form-control-lg mb-3'
            autoFocus
        />
        <button onClick={handleQuizSubmit} className='btn btn-success btn-lg w-100'>Submit</button>
        <p className='text-muted mt-3'>Question {quizIndex + 1} of {quizOrder.length}</p>
    </div>
)}
```

Figure 52 - React Code for 'Quiz' Game State: Paired Associate Learning

In the 'quiz' game state, the user is presented with a random stimulus word from one of the stimulus-response pairs. From this, the user must aim to input the correct response to the stimulus.

```
// Creates 6 random numbers (index) to quiz the user, where each index corresponds to a stimulus-response pair
const generateQuizOrder = (pairs) => {
    const indices = [];
    for (let i = 0; i < 6; i++) {
        indices.push(Math.floor(Math.random() * pairs.length));
    }
    return indices;
}
```

Figure 53 - React Code for generating the question order for the 'quiz' state

In this code, six random numbers are generated, where each number corresponds to an index in the stimulus-response pairs. After all pairs are displayed to the user in the 'slideshow' state, the solution navigates the user to the 'quiz' state of the game, where they are given a stimulus to which they must input the correct response. The function will take in an array of pairs, and an empty array 'indices' is implemented to store the six randomly generated numbers inside the for loop. Each randomly generated number is added to the array 'indices', which will be used to choose which six stimulus words are displayed to the user. Six numbers are generated as six pairs are initially shown to the

user, at times, the same stimulus will be shown to the user as the repetition of the stimulus can aid in memory recall.

```
const handleQuizSubmit = () => { // When user enters their answers to the stimulus
    const correctResponse = pairs[quizOrder[quizIndex]].response.toLowerCase(); // Correct response for the stimulus prompt is retrieved and converted to lowercase
    const userAnswer = userInput.trim().toLowerCase(); // User's answer => Whitespace is trimmed and converted to lowercase

    const isCorrect = userAnswer !== '' && userAnswer === correctResponse; // Boolean to check if user answer is correct

    if (isCorrect) {
        setScore(prev => prev + 1); // Increment score if answer is correct
    }

    setInput('');

    if (!isCorrect || quizIndex + 1 >= quizOrder.length) { // If the answer is wrong OR question length is reached
        sendScoreToBackend(userScore + (isCorrect ? 1 : 0)); // Score is submitted to backend
        setGameState('result');
    } else {
        setQuizIndex(prev => prev + 1); // Else if not finished, go to next question
    }
}
```

Figure 54 - React code to handle user submission

The above code conveys what takes place after the user submits their answer for the stimulus presented to them in the ‘quiz’ state. The correct response for each of the stimulus words is retrieved, using ‘.response’ which accesses the value stored in the ‘response’ key, and stored in the array ‘correctResponse’ from which it is converted to lowercase. Moreover, the user input is also stored in an array ‘userAnswer’ where potential whitespace is removed, and the answer is also converted to lowercase to aid with verification. The user’s answer is checked to see if it’s not empty and if it matches one of the responses in the ‘correctResponse’ array. If it’s true for both scenarios, then the user’s score increments, and the input field is cleared regardless of whether the answer was correct or not. However, if the answer is not correct or the question length is reached, the user’s score is submitted to the backend, and they are navigated to the Results Page.

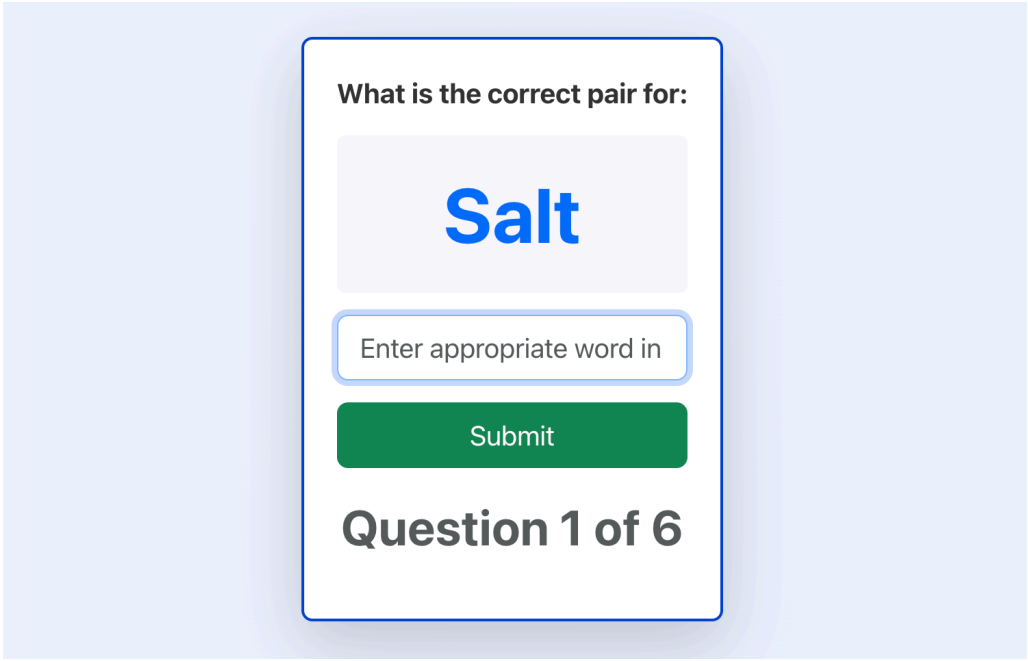


Figure 55 - The ‘Quiz’ Game State: Paired Associate Learning

Developing from Figure 34, which conveys the code for the 'quiz' game state, the application displays the stimulus from the pair so that the user may recall the correct response.

5.5. Self-Ordered Search ("Find The Diamond!" Game)

5.5.1 Implementation

The Self-Ordered Search aims to assess the cognitive domain of 'Executive Functioning'. Named "Find The Diamond!" to create a fun and engaging environment for users, a set of boxes will be presented, and there will be a target item (which is a diamond in this game), which is hidden underneath one of the boxes. The user will click the boxes until they find the diamond, however the diamond doesn't appear in the same box which will enable the user to utilise their memory and recall the boxes which previously contained the diamond so that they avoid selecting it again. To aid with this, a three-life system has been implemented so that if the user selects a box which previously contained the diamond, they will lose a life and if they lose all three lives, the game ends.

Like the Digit Span Test and Paired Associate Learning, this game contains similar features, such as the 'start' and 'result' game states.

5.5.1.1 Game States

```
const [gameState, setGameState] = useState("start");
```

Figure 56 – 'useState' hook for initial game state

Similarly, seen in previous games, the React useState hook is utilised to set the initial game state to 'start'. The game states for the Self-Ordered Search ("Find The Diamond!" game) are 'start', 'playing', 'result' and 'completed'.

```
if (gameState === "start") {
  return (
    <div className="d-flex align-items-center justify-content-center vh-100 bg-gradient-to-br from-blue-100 to-green-100">
      <div className="card text-center p-5 shadow-lg">
        <div className="d-flex justify-content-center mb-4">
          <Gem size={64} className="text-primary" />
        </div>
        <h2 className="display-5 mb-3 text-primary fw-bold">Find The Diamond!</h2>
        <i>
          <h2 className="display-5 mb-3 text-primary">
            Can you remember where the diamonds were?
          </h2>{" "}
        </i>
        <p className="fs-5 mb-4">
          1. Click on the boxes to find the hidden diamond. <br/>
          <br/>2. If you're wrong, the box turns red! 🟥  <br />
          <br /> 3. Once you find it, a new diamond will be hidden! <br/>
          <br/>4. <u>Don't click a box where a diamond was hidden before</u> – or you'll lose a life! 💔<br/>
          <br />5. Test your memory and become a true Diamond Master! 💎
        </p>
        <button onClick={startGame} className="btn btn-success btn-lg">
          Start Game
        </button>
      </div>
    </div>
  );
```

Figure 57 - React code for 'start' game state

Like the other games, the Self-Ordered Search (“Find The Diamond!” game) contains a start page conveying instructions on what the user should expect as well as a ‘Start Game’ button for the application to navigate the user to the ‘playing’ game state.

5.5.1.2 Box Generation

```
const getBoxCount = (lvl) => Math.min(3 + (lvl - 1) * 2, 9); // Getting no. boxes for each level: Level 1: 3 boxes, Level 2: 5 boxes, Level 3: 7 and Level 4: 9

const generateBoxes = (count) => { // Array of box objects
  return Array.from({ length: count }, (_, index) => ({ // Creates an array with length of 'count'
    id: index, // ID to make the boxes unique
    isRevealed: false, // Boolean to check if box has been clicked
    wasChecked: false, // Boolean to check if user has interacted with it
  }));
};
```

Figure 58 - React Code for Box Generation: Self-Ordered Search

There will be a specific number of boxes for each level, to ensure the user utilises their memory more as the game progresses. For this game to function accordingly, boxes will need to be generated to hide the target item (the diamond). To do this, an array is created with a specific length, and it will contain features such as ID so that each box is unique as well as Boolean values for whether the box has been clicked or if it was interacted with.

```
<div
  className={`row g-3 mb-4 ${boxes.length <= 4 ? "row-cols-2" : "row-cols-3"}`}
>
  {boxes.map((box) => (
    <div key={box.id} className="col">
      <div style={{ paddingBottom: "100%", position: "relative" }}>
        <button // Button appearance is different based on if box is revealed, wrong or has diamond
          data-testid={`box-${box.id}`}
          onClick={() => handleBoxClick(box.id)}
          disabled={isShuffling}
          style={{
            position: "absolute",
            top: 0,
            left: 0,
            width: "100%",
            height: "100%",
            fontSize: "2rem",
            borderWidth: "5px",
            borderStyle: "solid",
            borderColor: "#aaa",
          }}
          className={`btn
            ${
              box.isRevealed
                ? box.id === diamondPosition
                  ? "btn-success"
                  : "btn-danger"
                : "btn-outline-secondary"
            }`}
        >
          {box.isRevealed ? ( // To display either diamond or X icon
            box.id === diamondPosition ? (
              <Diamond className="text-white" size={32} />
            ) : (
              <X className="text-white" size={32} />
            )
          ) : null}
        </button>
      </div>
    </div>
  ))}
</div>
```

Figure 59 - React Code snippet of ‘playing’ game state

The above code is an extract of the ‘playing’ game state, which conveys the rendering of the interactive boxes. The styling of each box is conveyed in the lines ‘style={{…}}” and it changes based on whether a box has already been clicked by the user or if it matches the position of the diamond through ‘box.id === diamondPosition’. If the diamond position matches the box ID, the boxes colour will change to green (‘btn-success’) with a diamond icon conveyed in the middle of the selected box, and if not it will change to red (‘btn-danger’) with an ‘X’ icon in the middle of the box, while hovering over each box gives it a grey colour (‘btn-outline-secondary’).

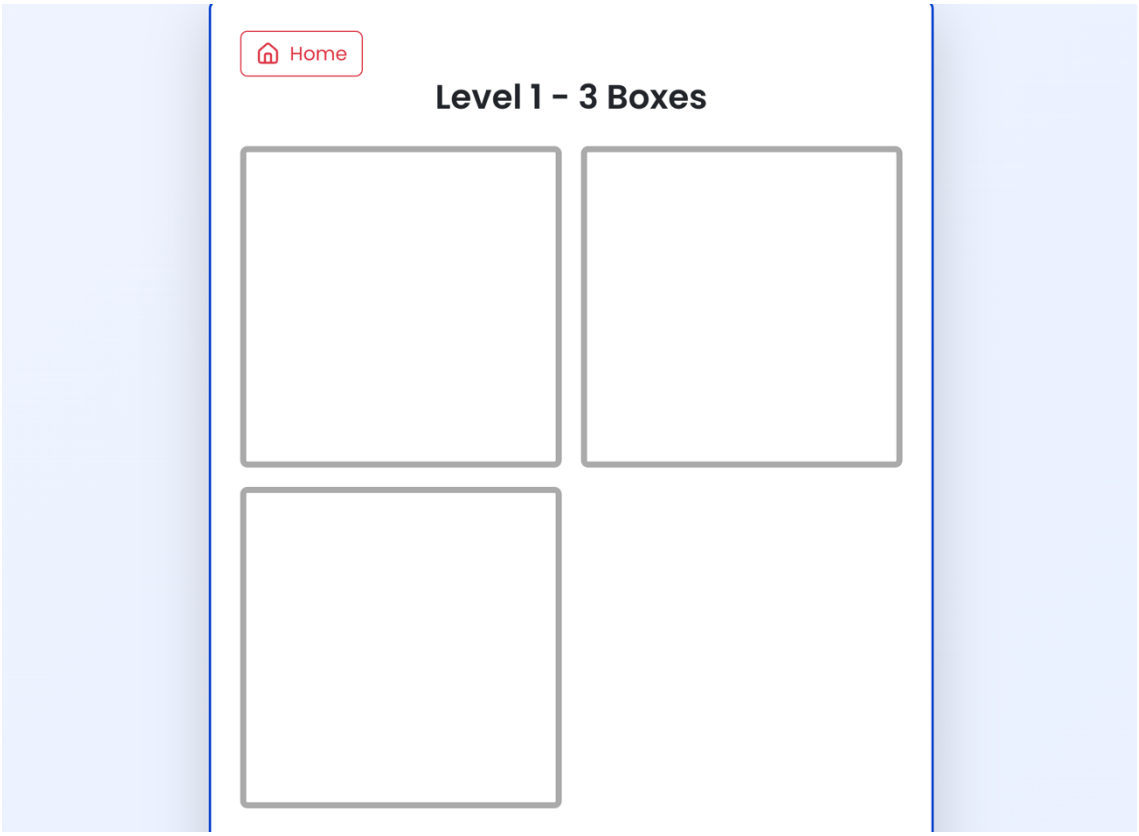


Figure 60 - Boxes: Self-Ordered Search

For the first level, users are presented with three boxes with the target item (a diamond) hidden underneath one of the boxes. A bold grey border was used for visual purposes, aiding users with visual impairment by making the boxes clearer.

5.5.1.3 Diamond Placement

```
const alreadyCorrectInSameLevel = successfulBoxes.some(b => b.id === boxId && b.level === level); // Checks if user found the diamond in the SAME BOX in this level
if (alreadyCorrectInSameLevel) { // If user clicks on the same box again
  setLives(prev => {
    const remaining = prev - 1;
    if (remaining <= 0) {
      setGameState('result');
    }
    return remaining;
  });
  return;
}

if (boxId === diamondPosition) { // If user finds the diamond
  const newFound = found + 1;
  const updatedSuccess = [...successfulBoxes, { id: boxId, level }];
  setFound(newFound);
  setSuccessfulBoxes(updatedSuccess);

  const levelUp = newFound % 3 === 0; // Update score for found diamond
  const newLevel = levelUp ? Math.min(level + 1, MAX_LEVEL) : level;

  if (newLevel > level) { // Update level and max level reached
    setLevel(newLevel);
    setMaxLevelReached(prev => Math.max(prev, newLevel));
  }

  setIsShuffling(true);
  setTimeout(() => {
    setupLevel(newLevel, updatedSuccess);
    setIsShuffling(false);
  }, 800); // There is a brief pause before next phase is loaded
}
};
```

Figure 61 - React Code for Diamond Placement: Self-Ordered Search

The aim of the function above is to check if the user has interacted with the same box that previously contained the diamond, which is carried out by checking the box's ID, 'boxID'. If the user clicks on the same box again, the number of lives they have will be deducted by 1, and if they reach zero lives, the game state will change to 'result'.

However, if the user clicks the correct box, with the diamond hidden, their score will increase by 1, and the selected box will be added to the array of previously successful boxes.

The level will increase every time the user finds 3 diamonds, and interaction with boxes will be temporarily disabled while the next level is being loaded.

5.5.1.4 User Interaction

```
const handleBoxClick = (boxId) => { // If game is setting up the next level, clicking boxes is disabled
  if (isShuffling) return;

  setBoxes(prev =>
    prev.map(box =>
      box.id === boxId ? { ...box, isRevealed: true, wasChecked: true } : box // Reveals the clicked boxes
    )
  );
```

Figure 62 - React Code for User Interaction: Self-Ordered Search

There is an initial check to see if the box can be clicked by the user, 'isShuffling' is initialised to false to prevent interaction when there is a change in levels. If the Boolean variable is found to be true, the function returns early, so that the user cannot interact with the boxes while it's changing levels.

However if the Boolean variable is false, the state of the clicked boxes is updated using 'setBoxes()'. Utilising '.map()', the function maps all boxes and sets its states

'isRevealed' and 'wasChecked' to true, to convey that the box has been interacted with and a diamond or X icon will be displayed inside of it.

```
    const alreadyCorrectInSameLevel = successfulBoxes.some(
      (b) => b.id === boxId && b.level === level
    ); // Checks if user found the diamond in the SAME BOX in this level
    if (alreadyCorrectInSameLevel) {
      // If user clicks on the same box again
      setLives((prev) => {
        const remaining = prev - 1;
        if (remaining <= 0) {
          setGameState("result");
        }
        return remaining;
      });
      return;
    }

    if (boxId === diamondPosition) {
      // If user finds the diamond
      const newFound = found + 1;
      const updatedSuccess = [...successfulBoxes, { id: boxId, level }];
      setFound(newFound);
      setSuccessfulBoxes(updatedSuccess);

      const levelUp = newFound % 3 === 0; // Update score for found diamond
      const newLevel = levelUp ? Math.min(level + 1, MAX_LEVEL) : level;

      if (newLevel > level) {
        // Update level and max level reached
        setLevel(newLevel);
        setMaxLevelReached((prev) => Math.max(prev, newLevel));
      }

      setIsShuffling(true);
      setTimeout(() => {
        setupLevel(newLevel, updatedSuccess);
        setIsShuffling(false);
      }, 800); // There is a brief pause before next phase is loaded
    }
  };
```

Figure 63 - React Code for User Interaction (continued)

Further developing from Figure 55, the above code further conveys what will take place in the game when the user clicks a box, such as changes in lives, score and game state.

The function 'alreadyCorrectInSameLevel', if the current box being clicked had previously contained a hidden diamond which the user successfully found. If the box has had already been clicked and revealed a diamond earlier in the level, the number of lives decrement and if the number of lives goes to 0, the game state will change to 'result', navigating the user to the Results page.

If the selected box matches the position of the diamond, in 'boxId === diamondPosition', the user's score is incremented, and the selected box is added to an array 'successfulBoxes'. The code also checks if all three diamonds are found in each level, if it's found to be true, the game progresses to the next level. However, if the maximum level is reached, the user gets directed to the 'result' page.

Moreover, the code also implements a pause by disabling interactions with boxes during transitions of levels, so that a new level can be set up.

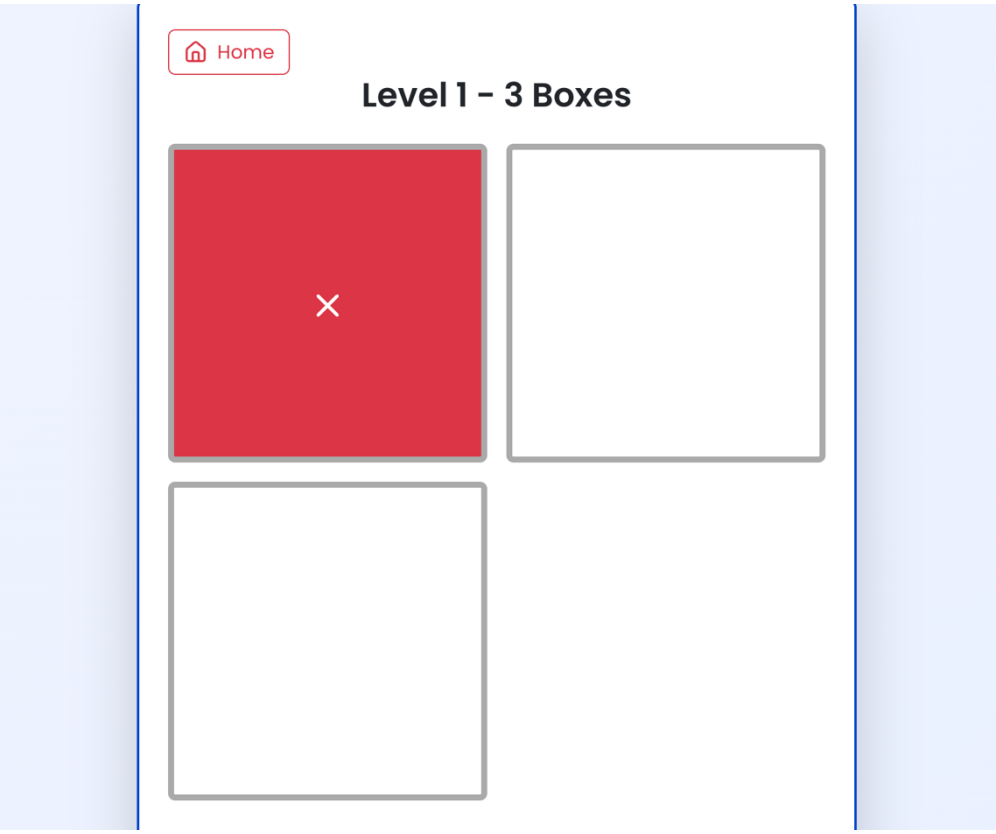


Figure 64 - User Interaction: Self-Ordered Search

5.6. Readability and Maintainability

Readability and maintainability were emphasised throughout the implementation process of the application, additionally to also aid with potential future improvements which was attained through consistent programming practices.

5.6.1 Comments and Function Names

Throughout all pages within the application, function names clearly conveyed their role in the code, in addition the utilisation of comments inside the functions aided in breaking down visually complex code into simpler terms, giving the reader a clear understanding of the purpose of each line of code. Moreover, the reusability of function names throughout different files in the coding solution assisted in identifying vital roles in making the application perform as intended.

5.6.2 Indentation and Coding Conventions

To maintain both programming readability as well as a clear structure across the application, indentation and coding conventions were utilised. In the backend of the solution, the Python code written strictly adhered to the PEP8 guidelines: such as naming conventions, spacing and formatting across all Python files. In the frontend of the application, CamelCase formatting was employed for both variable and function names to adhere to common JavaScript conventions. Additionally, indentation was used consistently throughout all JavaScript files to aid with development ensuring that coding hierarchy and formatting was understandable.

5.7. Challenges/Issues Encountered

Throughout the development of this complex solution, several obstacles were faced. The table presented in **Appendix D** conveys the challenges faced and how they were tackled and resolved.

Chapter 6: **Testing**

Testing is a vital step in ensuring that specific components within the application function as intended. Testing methods such as Unit Testing and User Acceptance Testing were utilised to test if individual components within the application were working as expected, the presence of communication between the frontend and backend, as well as assessing whether the solution meets the needs of the primary stakeholder.

To aid with testing, changes were made to the structure of the application where files such as the 'babel-config-js' file, which uses both '@babel/preset-env' and '@babel-preset-react', were created to enable the conversion to an older and more compatible version of JavaScript. Additionally, the 'jest-config.js' file was created to utilise 'babel-jest' as well as using 'identity-obj-proxy' to handle CSS and 'transformIgnorePatterns' to aid with using axios to be babel-transformed. Mock files, stored in the folder '__mock__', were also created to avoid import errors when tests are run.

6.1. Unit Testing

Unit Testing aims to test components individually in the application. It will ensure that logic from the React components function as intended: such as the inputs for the login and signup forms, buttons within the application as well as the cognitive games. The table shown in **Appendix E** convey the unit tests carried out. Tests were executed and modified ensuring that they passed and gave the expected output in the terminal.

```
Test Suites: 5 passed, 5 total
Tests:       26 passed, 26 total
Snapshots:   0 total
Time:        3.008 s
Ran all test suites related to changed files.
```

Figure 65 - Expected Terminal Output for all unit tests

While conducting Unit Tests for the appropriate components, edge case testing was found to be not feasible when testing the Self-Ordered Search game. This was because the game relied on internal randomisation of the placement of diamonds hidden behind the boxes, which additionally made it not feasible to track life counts and diamond count. Unit tests conducted for the Self-Ordered Search were more focused on the transitional elements of the game, such as transitions from game states.

6.2. User Acceptance Testing

User Acceptance Testing is a significant form of testing, especially for this solution. With this solution being tailored towards individuals suffering from dementia, it was vital to receive feedback about the solution. Microsoft Teams and Zoom meetings were conducted with medical students and doctors, where they were able to take control and traverse through the web application. They were asked to play through each of the games as well as traverse through different pages of the application, as well as provide useful feedback on areas that needed improvement. Areas of improvement were highlighted and amended in **Appendix E**, and alterations to the UI were made to be more

suitable for the primary stakeholder. Overall, the medical students and doctors who assessed the application deemed it to be appropriate for the target audience due to the simplicity of the design and the gameplay experience.

Chapter 7: **Evaluation**

7.1. Requirements and Testing Evaluation

Functional and non-functional requirements were set out at the initial stages of implementation and an updated version of the requirements were created during the mid-stages of development, which was followed throughout the development process. Testing revealed that all the functional and non-functional requirements listed were successfully fulfilled.

Unit tests were conducted, utilising the React Testing Library, on the core React components within the application which includes 'Login.jsx', LoginSignup.jsx', 'DigitSpan.jsx', 'PairedAssociateLearning.jsx' and 'SelfOrderedSearch.jsx'. The React components rendered successfully, implemented edge case scenarios were handled accordingly and the components responded appropriately to actions carried out by the user. The unit tests confirmed that the functional requirements were successfully achieved.

However, minor issues were detected while carrying out the unit tests, such as mock setups for fetch requests being incorrect and displayed warnings in the terminal due to asynchronous actions taking place. Fortunately, through iterative rectification of the unit tests, the issues detected were resolved. Additionally, API endpoints developed in Flask were also tested in the unit tests through interactions in the frontend, such as submitting the scores from a game or login/signup processes, conveying clear communication between the frontend and backend.

User Acceptance Testing was conducted through Microsoft Teams and Zoom sessions with medical students and doctors, who have experience in dementia care. They assessed the functionalities of the games as well as the usability of the web application by taking into account the cognitive abilities of individuals with dementia. Overall, feedback given was positive, conveying the suitability of the games for patients with dementia however minor proposals to adjust the user interface such as font size and structural layout of the progress page.

Both tests conveyed that functional and non-functional requirements were successfully met.

7.2. Stakeholder Evaluation

Stakeholder Evaluation was a vital step in the development of this solution, as it ensured the application met the needs of its target audience. Carried out through informal discussions on WhatsApp and Discord calls, medical students with experience in dementia care significantly aided in the development process by providing useful feedback on how the application can be altered and enhanced to be more appropriate for individuals with dementia.

7.2.1. Mid-Development Feedback

During the development process of the solution, a mid-development version of the application was presented to the medical students, where they highlighted the significance of simplicity and a minimalistic design. Some examples of feedback given was the removal of animations utilised as well as increasing the font-size and button size,

in order to prevent cognitive overload among dementia patients. As a result of this, the appropriate adjustments were made to the design of the application.

7.2.2. Post Development Feedback

After having implemented the core features as well as additional features in the application, the final version of the solution was presented to the same group of medical students. They were satisfied with the overall application, conveying how the games were engaging and the simplistic design of the application. Minor suggestions were made with the structural layout of the progress page, conveying that the design should be visually clearer for users as well as the addition of a visual aid on how to operate the games. This feedback has been helpful and will be considered as areas for future improvement.

7.3. Evaluation against Aims and Objectives

As highlighted earlier, the aim of the solution is to create a user-friendly environment tailored towards individuals suffering from dementia, where they can take part in numerous engaging games to improve overall cognition. This was achieved, a full-stack web application was created using React.js and Python (Flask) which included all the necessary pages as well as the cognitive games which were The Digit Span Test (named 'Mathemagician' in the application), Paired Associate Learning (named 'Pair Up!') and Self-Ordered Search (named 'Find The Diamond!').

The following objectives were set out at the initial stages of development:

Objective ID	Objective	Outcome (Achieved/Not Achieved)
OBJ1	Conduct a Comprehensive Literature Review on how my application may benefit the user	**Achieved:** An extensive Literature Review was conducted to not only understand how the solution can be tailored toward the target audience, but also to be better educated on dementia: highlighting the effects of cognitive decline on an individual, various cognitive domains and how each one is treated through various tests and the effects of certain illnesses on dementia.
OBJ2	Conduct a Comprehensive Data Gathering process on how my application can be better tailored towards dementia patients	**Achieved:** Both surveys and interviews were conducted with medical students and doctors who have experience with dementia care. The feedback given aided in developing the solution, gaining an insight into how effective cognitive games are for patients with dementia and a general idea of how to design the application.

OBJ3	Gain a perspective of dementia patients in order to understand new ways to integrate certain elements to the application	**Achieved:** Both mid-development and post-development feedback provided further support in understanding how the application can be tailored towards individuals with dementia. The feedback allowed for there to be changes in the UI such as the size of fonts and buttons, as well as simple navigation through the application.
OBJ4	Design a web application that contains cognition assessments to improve the user's cognition, it will include the following pages: 'Login/Register' Page, 'Dashboard' Page, A 'Progress' Page, 'Games' Page (listing all different types of games for the user to play)	**Achieved:** A full-stack web application was created using React.js for the frontend and Flask for the backend. The 'Login/Register' Page provided features for account registration and logging into the system, the 'Dashboard' Page allowed users to navigate to the 'Progress' and 'Games' Page, The 'Progress' Page showed user-specific recent and highest scores on each of the games and the 'Game' Page allowed users to select each of the games and give a brief overview on what to expect when taking part in each of the games.

Table 5 - Evaluation of Objectives Table

7.4. Legal, Social, Ethical and Accessibility issues

7.4.1. Legal Issues

Considerations around legality were taken into account when carrying out specific steps in the development of this solution. During the data gathering process as well as the User Acceptance Testing, participant anonymity was ensured with responses in surveys and interviews being kept confidential. In accordance with GDPR, no sensitive or personal data was stored.

In addition, secure user authentication was handled accordingly by the frontend and backend of the application through the utilisation of password hashing, JWT tokens and password strength validation in the 'Login/Signup' page.

7.4.2. Social Issues

The primary social challenge this solution aims to tackle is to provide support to individuals suffering from dementia through interactive and engaging cognitive games. Dementia, being an illness with no definitive cure, has a huge negative impact on not only the affected individual, but their family as well as their carers in certain cases. This solution aims to improve cognitive health as well as the user's overall quality of life through easy, engaging, non-pharmaceutical cognitive exercises.

This solution aids in mitigating the need for the carers and family to actively monitor or encourage the user to participate in regular cognitive exercises. Overall, the application will not only provide support to the user but also to their surrounding parties involved.

7.4.3. Ethical Issues

Issues around ethics were addressed in the development of this solution, specifically in both the comprehensive data gathering process as well as the User Acceptance Testing. Doctors and medical students with experience in dementia care were chosen as participants for surveys, interviews, mid-development and post-development feedback as well as for the User Acceptance Testing as opposed to individuals suffering from dementia or other forms of cognitive impairments, as this would cause a lot of unnecessary stress and anxiety on the vulnerable individuals and taking into consideration their difficulties in their understanding of consent, this approach would be deemed unethical. As a result of this, collecting data from individuals experienced in dementia care allowed invaluable information to be gained ethically.

7.4.4. Accessibility Issues

Accessibility was a vital area of focus in the development process of the solution, considering the primary stakeholder of the application. Through both independent research as well as feedback given by doctors and medical students, the overall design of the application aimed at reducing cognitive overload by incorporating clear text, simple navigation and layout as well as labelled buttons. Both mid-development and post-development feedback highlighted areas of improvement in the User Interface, such as alterations to the structural layout of the Progress page, to help reduce potential cognitive overload.

7.5. Usability, Reliability, Maintainability and Security

Evaluation of Usability, Reliability, Maintainability and Security is carried out to ensure the developed solution has a strong foundation in vital software quality areas.

Assessed Aspect	Evaluation
Usability	The main aim of this application was to create an engaging, interactive environment for dementia users to take part in cognitive games to improve their overall cognitive wellbeing. Features implemented into this solution includes simple navigation, simplistic and engaging design, readable fonts and buttons, and consistent design throughout all pages in the application. With the aid of feedback given during development process as well as in the User Acceptance Testing, changes were made to specific pages in the application to make it more tailored toward the target audience. Such as alterations in the structural layout of the Progress page and the changes to the font size in the ‘Start’ screen for each of the games. However, a potential screen reader and audio feedback to convey errors that have taken place. This would aid in providing support to users who also suffer from visual impairment. Taking into account the majority of users using the application would be elderly individuals.
Reliability	It was vital to test that interaction with the application was smooth and no disruptions would occur. Unit tests were carried out to test several

	core React components in the application such as 'Login.jsx', 'LoginSignup.jsx', 'DigitSpanjsx', 'PairedAssociateLearning.jsx' and 'SelfOrderedSearch.jsx' to check that components within the application work and there is a API smooth communication between the frontend and backend. In addition, User Acceptance Testing also aided in ensuring the games functioned as expected. During the testing process, minor issues were detected in some of the unit tests which were quickly resolved through iterative debugging. For future improvements, end-to-end testing could be implemented using Selenium to validate user workflows across the frontend and backend.
Maintainability	Maintainability was a core part of the development process of this application, through the utilisation of React components as well as the separation of frontend and backend code into their respective folders. The Python (Flask) Code adhered to PEP8 code format standards and regular commenting was utilised throughout the solution to aid with potential developers with understanding the system. For future improvements, the utilisation of more unit tests for smaller components within the application could aid in further increasing maintainability, reducing the risks of potential bugs arising from potential additions or development. As well as the addition of a neural network that analyses the past scores of users throughout all the games in the solution and recommends the most helpful game for the user to take part in.
Security	Security was addressed in the development process through secure user authentication. In the backend, password hashing was utilised alongside JWT tokens for session authentication for the login and registration process. In the frontend, password strength validation was used to ensure a strong password is used during registration. To access user-specific data such as scores from the games, secure tokens were used and were fetched and displayed onto the Progress page. For future improvements, the implementation of HTTPS enforcement can provide an additional layer of security further ensuring that the user's data is encrypted.

Table 6 - Table for Evaluation of Usability, Reliability, Maintainability and Security

Chapter 8: **Conclusion**

8.1. Evaluation of Cognitive Game-based Web Application

The aim of this project was to build an engaging cognitive game-based application for dementia patients where they can take part in various games aimed at targeting different cognitive domains such as 'Complex Attention', 'Executive Functioning' and 'Learning and Memory'.

As a result of this, a user-friendly interface was created utilising React.js and the application of additional tools such as Bootstrap and libraries like 'lucide-react' for design purposes, ensuring a simplistic and clear layout for the primary stakeholder. In addition, a backend was developed using Flask, which contained fully functional REST APIs handling various tasks such as user authentication, score handling for the cognitive games as well as fetching user progress. Each of the games in the application was developed with the aim of becoming more challenging as the user progresses as well as being engaging and forcing the users to utilise their memory and problem-solving skills. Additional features were implemented into the application to make it both more engaging as well as dementia-friendly such as motivational quotes, a trophy system, video tutorials on how to utilise the games as well as helpful prompts to aid users through navigation.

Overall, mid-development and post-development feedback as well as User Acceptance Testing, conveyed that the developed solution not only met the proposed objectives but also met the standards for a dementia-friendly application. The solution was accessible, scalable and has shown to have potential to be utilised in potential clinical research use.

8.2. Advanced Technical Skills

Several advanced problem-solving skills were utilised throughout the development process of this solution.

8.2.1. Technical Complexity

Technical Complexity	Justification
REST API integration	A vital part of the development process was to ensure there was smooth communication between the frontend and the backend of the solution, utilising REST API principles such as HTTP methods like GET and POST to retrieve and send data, creation of clear API endpoints from score submissions to user authentication as well as using JSON format to send and receive data.
Implementation of Self-Ordered Search ('Find The Diamond!' Game)	Application of multiple dynamic states aided in tracking diamonds found by the user as well as tracking boxes that have already been selected by the user for decrementing the life count feature.
User Authentication	Implemented a secure user authentication system for login and signup process, with elements such as password hashing as well as JWT tokens for session authentication to ensure security in the system. In addition, edge case handling was

	also implemented for duplicate emails or weak password validation.
User score fetching in Progress Page	Developed a method for user-specific score retrieval, where data is sent through an authenticated request from the frontend to the backend via the user's token, which then allows the user's score to be fetched separately for each of the games from the database to be displayed onto the Progress Page.

Table 7 - Technical Complexity Table

8.2.2. Technical Creativity

Technical Creativity	Justification
Trophy System and Motivational Quotes	Trophy system and motivational quotes were implemented with user experience in mind. With the intention of enhancing engagement, these features will aid with motivating regular use of the application for the user. The trophy system will allow the user to gain a sense of accomplishment when attaining high scores in the cognitive games.
Dementia-Friendly User Interface	A simplistic and clear layout was utilised throughout all pages of the application. Text fonts were readable and clear, buttons were big, and light colours were added to create a visually user-friendly environment for the target audience. Prompts were utilised throughout pages to aid with navigation and a minimalistic design was added to mitigate the likelihood of cognitive overload.
Slideshow Utilisation in Paired Associate Game ('Pair Up!' Game)	Slideshow mechanism was employed into this game to mimic the Paired Associate Learning assessments given to dementia patients to complete in a medical environment, to be completed in a more comfortable environment. Using this mechanism, allows users to push their memory skills to the limits to try and recall stimulus-response pairs from memory using the slideshow.
Difficulty Implementation in Paired Associate Learning Game ('Pair Up!' Game)	Taking into account users with various levels of cognition will be using this application, a difficulty system was implemented into this game. The game gets progressively challenging based on the complexity of the words added into each level, allowing users with no experience in cognitive assessments taking part in 'Easy' mode and more experienced users selecting the 'Medium' or 'Hard' difficulty.

Table 8 - Technical Creativity Table

8.2.3. Technical Challenges

Technical Challenges	Justification
User Authentication	Having allocated this task closer to the end of the development process, user authentication was seen a huge obstacle. This process required numerous implementation steps such as utilising JWT tokens for session authentication, password hashing for security, as well as the addition of edge cases to handle duplicate emails, weak password validation and invalid credentials.
Implementation of Cognitive Games	The process of developing each of the cognitive games was seen as a huge challenge. Alongside the development process, research had to be carried out on what the games were through medical papers online as well as through informal discussions with medical students, it was necessary to also further develop current knowledge on React to aid with development of the games such as learning complex state management. In addition, the implementation of each of the games required different logic for example the Digit Span Test contained a sequence of numbers that would get longer through each iteration and the Paired Associate Learning contained a slideshow mechanism displaying the stimulus-response pairs to the user.
Learning Flask	Having no experience with programming in Flask, this was a huge challenge in my development process which had to be overcome as soon as possible near the early stages of development. Flask was seen as a primary choice of framework as it was suitable for this project due to its smooth integration with React.js. Some of the challenges that occurred were API integration, user authentication integration as well as managing appropriate response to requests made from the frontend.
Learning React Unit Testing	One significant challenge that occurred in the later stages of development was learning how to write unit tests using React.js. Having no prior experience with writing unit tests in React, key areas of struggle were creating mock API calls, testing different scenarios such as edge cases based on user interaction and wrong API responses.

Table 9 - Technical Challenges Table

8.3. Contributions to the Field

8.3.1. Utilisation of Solution in Medicine

The developed solution has the potential to be used in the field of dementia care and cognitive impairment for support purposes. By utilising existing cognitive assessments used to assess a patient's cognitive domains in a medical environment such as the Digit Span Test and Paired Associate Learning, medical professionals can conduct research

using this application by monitoring changes in cognition through various games in a non-medical environment.

The web application contains dementia-friendly features such as a minimalistic design, big buttons, clear text as well as simple navigation which makes it suitable for clinical research purposes where non-technical tools are required. In addition, overall cognitive performance can be tracked using the Progress page of the application, where user-specific scores are displayed for each of the games, which could allow medical professionals to observe changes in overall cognition.

Through further development of the application, more cognitive games can be implemented targeting additional cognitive domains enabling medical professionals to assess overall cognition to a greater extent.

8.3.2. Utilisation of Solution in Academia

Additionally, the developed solution can also be used in academic research, especially when studying topics such as cognitive training and impairment, where the solution provides an environment for users to assess different cognitive domains through games. The application is user-friendly and easy to use and can provide academic researchers with a useful environment to study cognitive training across different domains.

Moreover, the application can act as a foundation for the development of future cognitive game-based solutions, allowing academic researchers to not only utilise the current functionalities in the application but to also input additional features to make the application tailored toward their specific cognitive study, such as the implementation of other cognitive games to assess different cognitive domains like ‘Language’, ‘Perceptual-motor’ and ‘Social Cognition’.

The application can also act as an academic resource, aiding students seeking development in full-stack web development in the health-tech sector. The use of commenting throughout the solution as well as the combination of React.js for the frontend and Flask for the backend, enables students to learn vital skills in full-stack development such as API integration, state management and user authentication. Overall, the developed application can be utilised as a tool for both cognitive research and health-tech purposes.

8.4. Methodological Rigour and Project Management

In the initial stages of development, a Gantt Chart was created (**Appendix G**) providing a brief overview on the key milestones such as the ‘Project Definition’, ‘Draft Report’ and ‘Final Report’. These milestones provided a general idea of how long each of these major tasks should take as well as time allocation for vital subtasks within the development process such as learning Flask and implementing the cognitive games using React.js.

An Agile development approach was taken when developing the application, tasks were divided into development sprints. Sprints early in the development process were tailored towards creating the necessary pages for the application, including the implementation of the cognitive games. Sprints following this targeted on developing the backend of the application such as building REST APIs with user authentication implemented later to enable more time for gaining familiarity around programming with Flask. The final development sprints implemented were the testing stages, involving both unit testing as

well as User Acceptance Testing to ensure React components were functioning as expected.

The principles of agile methodologies were maintained through the development process of this project. Principles such as iterative development, flexibility and collaboration with medical students were heavily emphasised throughout the development of this solution. Iterative development was a vital part of building the solution, components were iteratively tested using unit tests and rectified through iterative debugging. Flexibility was crucial, specifically when huge obstacles arose, such as backend API handling in Flask and creating mock API calls through unit testing in React. Regular collaboration was crucial in ensuring this application was dementia-friendly, with regular post-development feedback highlighting areas for improvement in the application.

For future improvement, an Agile Scrum methodology should be employed. Although Scrum-based agile principles were followed during the development process, a Scrum methodology ensures that sprint reviews must take place after each sprint is carried out. In addition, by adhering to proper Scrum planning, core functionalities of an application such as user authentication should be handled in earlier sprints.

8.5. Future Development

Future development of this solution will include the addition of more games aimed at targeting a wider range of cognitive domains such as the inclusion of a Verbal Reasoning game aimed at targeting the cognitive domain of 'Language' (Brickman *et al.*, 2005), as well as a Corsi Block-Tapping test to assess the cognitive domain of 'Visuospatial short-term memory' (Vandierendonck *et al.*, 2004). Combining knowledge in full-stack web development with video game artificial intelligence, an adaptive difficulty tool can be implemented to aid with enabling the solution to adjust each of the game's difficulties based on the user's score. One method this can be achieved is through the development of an Artificial Neural Network (ANN), utilising intelligent agents to monitor user interaction and adjust the difficulty of the game accordingly as portrayed through a study carried out by Yannakakis et al. who used an artificial neural network which took in data such as the number of user interactions in order to alter the difficulties of games (Yannakakis and Hallam, 2009).

Moreover, the implementation of a Multilayer Perceptron (MLP) can aid in providing game recommendations based on past user performances such as time taken to complete a game and number of errors made by the user, from this data the MLP can recommend games for the user to play in order to improve their cognition in a specific domain.

As highlighted earlier, the inclusion of an Agile Scrum methodology can support in future development of this application, enabling a more effective development process through regular sprint reviews and the prioritisation of core features such as user authentication.

8.6. Summary

In summary, the proposed solution achieved the declared aims and objectives through the development of a fully functional, full-stack cognitive game-based application for patients with dementia. The application provides a dementia-friendly interface and engaging environment for users to take part in various progressively challenging cognitive games. By conducting an extensive study through the Literature Review as well as comprehensive data gathering through surveys and interviews with medical students

and doctors to gain a better understanding of the effects of dementia on everyday life and overall cognition, a solution appropriate for dementia patients was created. After an extensive testing process was carried out, the core components within the application functioned as expected such as the user authentication system and the cognitive game logic. Post-development feedback from medical students conveyed that the developed cognitive game application successfully met the standards for its target audience.

Through further development, the application can implement a neural network for game recommendations, wider range of cognitive games as well as an adaptive difficulty mechanism conveying its potential to be utilised as a solution in a clinical setting as well as an AI-driven game system to support medical professionals in monitoring overall cognition.

References

Andersson, M. *et al.* (2012) 'A population-based study on dementia and stroke in 97 year olds', *Age and Ageing*, 41(4), pp. 529–533. Available at: https://doi.org/10.1093/ageing/afs040.

Arvanitakis, Z., Shah, R.C. and Bennett, D.A. (2019) 'Diagnosis and Management of Dementia: A Review', *JAMA*, 322(16), p. 1589. Available at: https://doi.org/10.1001/jama.2019.4782.

Brickman, A.M. *et al.* (2005) 'Category and letter verbal fluency across the adult lifespan: relationship to EEG theta power', *Archives of Clinical Neuropsychology: The Official Journal of the National Academy of Neuropsychologists*, 20(5), pp. 561–573. Available at: https://doi.org/10.1016/j.acn.2004.12.006.

Butler, R. and Katona, C. (2019) *Seminars in Old Age Psychiatry*. Cambridge University Press.

Byers, A.L. and Yaffe, K. (2011) 'Depression and Risk of Developing Dementia', *Nature Reviews. Neurology*, 7(6), p. 323. Available at: https://doi.org/10.1038/nrneurol.2011.60.

Choi, H.J. *et al.* (2013) 'A Normative Study of the Digit Span in an Educationally Diverse Elderly Population', *Psychiatry Investigation*, 11(1), p. 39. Available at: https://doi.org/10.4306/pi.2014.11.1.39.

Dementia worry: a psychological examination of an unexplored phenomenon - PMC (no date). Available at: https://pmc.ncbi.nlm.nih.gov/articles/PMC5549110/ (Accessed: 15 November 2024).

van Dijk, P.T., Dippel, D.W. and Habbema, J.D. (1991) 'Survival of patients with dementia', *Journal of the American Geriatrics Society*, 39(6), pp. 603–610. Available at: https://doi.org/10.1111/j.1532-5415.1991.tb03602.x.

Gale, S.A., Acar, D. and Daffner, K.R. (2018) 'Dementia', *The American Journal of Medicine*, 131(10), pp. 1161–1169. Available at: https://doi.org/10.1016/j.amjmed.2018.01.022.

Harvey, P.D. (2019) 'Domains of cognition and their assessment', *Dialogues in Clinical Neuroscience*, 21(3), p. 227. Available at: https://doi.org/10.31887/DCNS.2019.21.3/pharvey.

Hugo, J. and Ganguli, M. (2014a) 'Dementia and Cognitive Impairment: Epidemiology, Diagnosis, and Treatment', *Clinics in geriatric medicine*, 30(3), p. 421. Available at: https://doi.org/10.1016/j.cger.2014.04.001.

Hugo, J. and Ganguli, M. (2014b) 'Dementia and Cognitive Impairment: Epidemiology, Diagnosis, and Treatment', *Clinics in geriatric medicine*, 30(3), p. 421. Available at: https://doi.org/10.1016/j.cger.2014.04.001.

Nasreddine, Z.S. *et al.* (2005) 'The Montreal Cognitive Assessment, MoCA: a brief screening tool for mild cognitive impairment', *Journal of the American Geriatrics Society*, 53(4), pp. 695–699. Available at: https://doi.org/10.1111/j.1532-5415.2005.53221.x.

Ninomiya, T. (2014) 'Diabetes mellitus and dementia', *Current Diabetes Reports*, 14(5), p. 487. Available at: https://doi.org/10.1007/s11892-014-0487-z.

Piolatto, M. *et al.* (2022) 'The effect of social relationships on cognitive decline in older adults: an updated systematic review and meta-analysis of longitudinal cohort studies', *BMC Public Health*, 22(1), p. 278. Available at: https://doi.org/10.1186/s12889-022-12567-5.

Raval, N.R. *et al.* (2022) 'An in vivo Pig Model for Testing Novel Positron Emission Tomography Radioligands Targeting Cerebral Protein Aggregates', *Frontiers in Neuroscience*, 16, p. 847074. Available at: https://doi.org/10.3389/fnins.2022.847074.

Sun, H. *et al.* (2022) 'IDF Diabetes Atlas: Global, regional and country-level diabetes prevalence estimates for 2021 and projections for 2045', *Diabetes Research and Clinical Practice*, 183, p. 109119. Available at: https://doi.org/10.1016/j.diabres.2021.109119.

Torian, L. *et al.* (1992) 'The effect of dementia on acute care in a geriatric medical unit', *International Psychogeriatrics*, 4(2), pp. 231–239. Available at: https://doi.org/10.1017/s1041610292001066.

Vandierendonck, A. *et al.* (2004) 'Working memory components of the Corsi blocks task', *British Journal of Psychology (London, England: 1953)*, 95(Pt 1), pp. 57–79. Available at: https://doi.org/10.1348/000712604322779460.

Yannakakis, G.N. and Hallam, J. (2009) 'Real-Time Game Adaptation for Optimizing Player Satisfaction', *IEEE Transactions on Computational Intelligence and AI in Games*, 1(2), pp. 121–133. Available at: https://doi.org/10.1109/TCIAIG.2009.2024533.

Appendix A – Interview Questions

1. Which cognitive domains are affected most by patients suffering from dementia?

2. Do you think patients would be interested in using games or applications to improve your memory?

3. Do you know if any patients have tried out or currently trying any methods to improve their overall cognition? If so, have you noticed any improvements? What challenges have they faced?

4. What do you hope this application achieves for patients with dementia?

5. Have patients used any applications or any other activities to improve their memory?

6. What type of cognitive-based activities or games showed the most benefits for dementia patients?

7. Do you have any foreseeable concerns or worries around using this application for dementia patients? Such as understanding instructions on each activity or how to make the best use of the application?

8. Would you say there are any ethical considerations when designing this application for vulnerable groups such as dementia patients?

Appendix B – Survey Questions

Healthcare Perspectives on Cognitive Games in Dementia Care

This short survey is part of a final year university project exploring the development of a cognitive game-based application to support memory and cognitive function in dementia patients. Your insights as a medical student or healthcare professional will help guide key design and functionality decisions. Responses are anonymous and will only be used for academic research.

Please rate your answers using a scale from 1 to 5, where:

- **1 = Not at all / Very Poor**
- **2 = Slightly / Poor**
- **3 = Moderately / Average**
- **4 = Very / Good**
- **5 = Extremely / Excellent**

Your responses are anonymous and will only be used for academic research purposes. Thank you for your time and valuable input.

How effective do you think cognitive games are in improving overall cognition in dementia patients?

1 2 3 4 5

How important would you say a simple user interface for this application is for patients to use?

1 2 3 4 5

Do you think visual aids are necessary to make this application easier to use for dementia patients?

1 2 3 4 5

How comfortable are patients with engaging in tasks that involve problem solving?

1 2 3 4 5

What would you rate your ability to concentrate and stay concentrated while completing regular tasks?

1 2 3 4 5

How well would you say patients can quickly process information and make judgements?

1 2 3 4 5

How well would you say it is for patients to pick up new skills or learn new information?

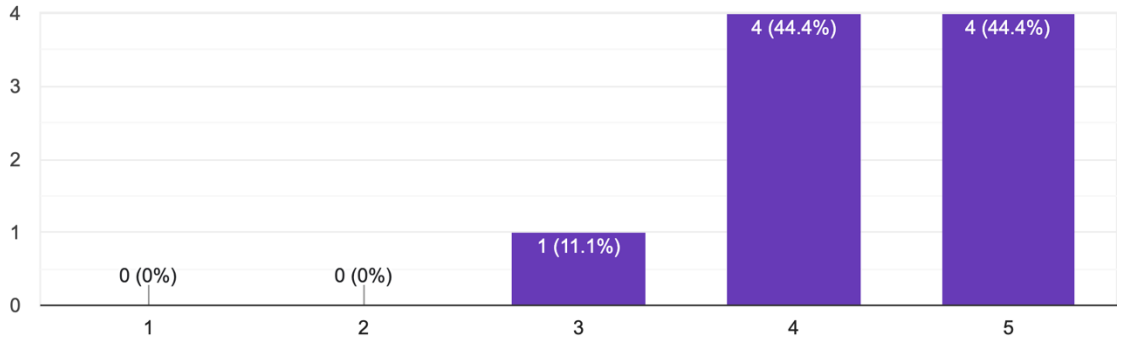
1 2 3 4 5

Submit Clear form

Appendix C – Survey Results

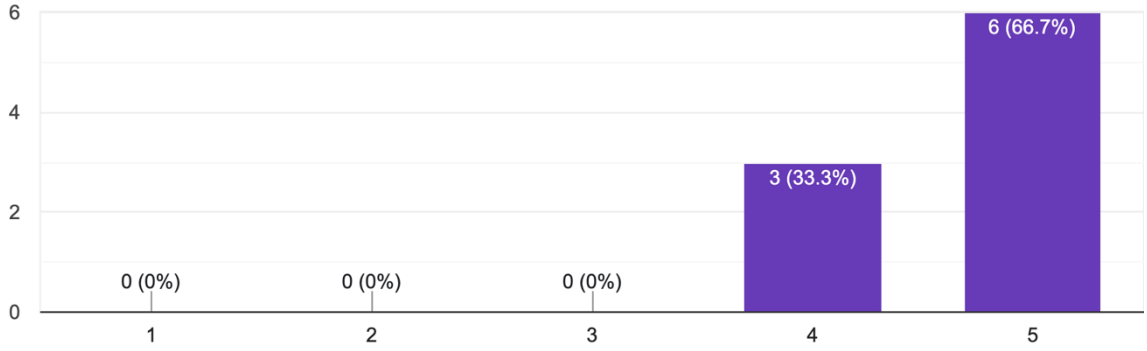
How effective do you think cognitive games are in improving overall cognition in dementia patients?

9 responses



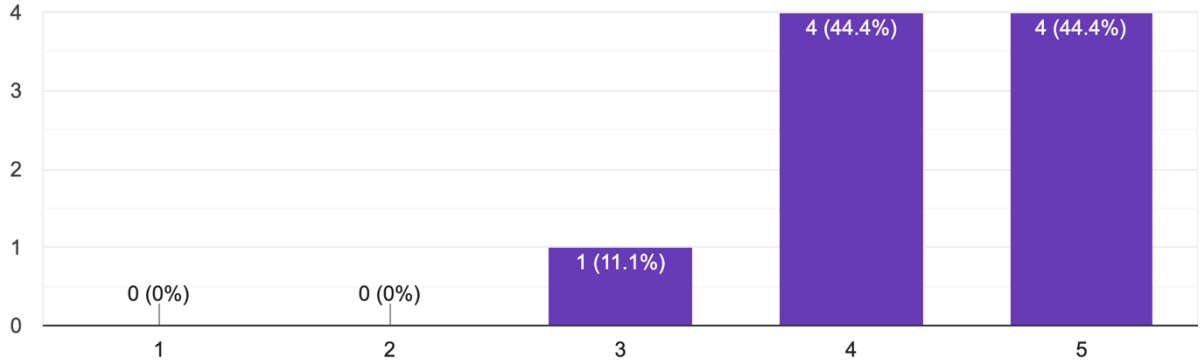
How important would you say a simple user interface for this application is for patients to use?

9 responses



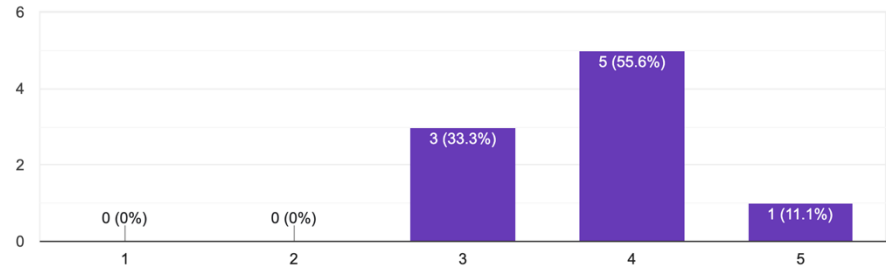
Do you think visual aids are necessary to make this application easier to use for dementia patients?

9 responses



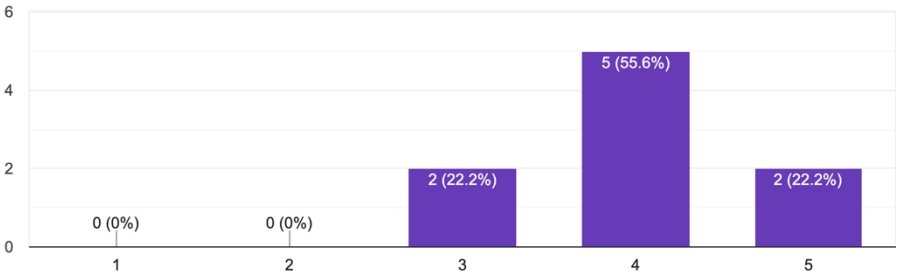
How comfortable are patients with engaging in tasks that involve problem solving?

9 responses



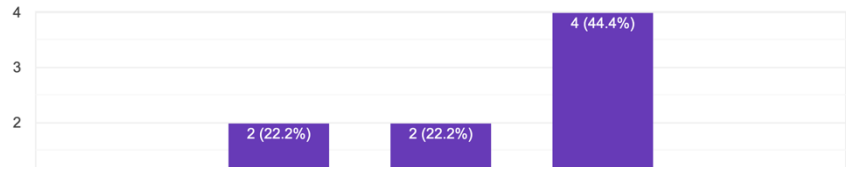
What would you rate your ability to concentrate and stay concentrated while completing regular tasks?

9 responses



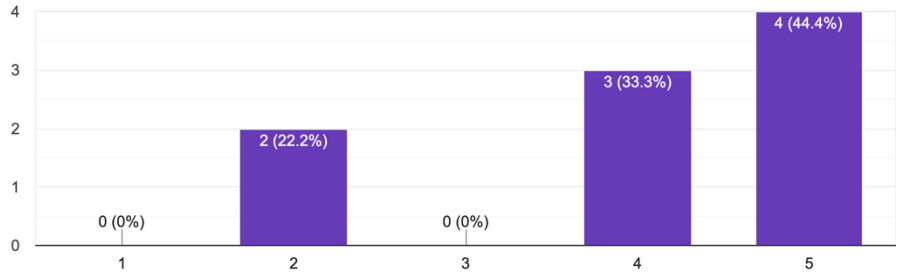
How well would you say patients can quickly process information and make judgements?

9 responses



How well would you say it is for patients to pick up new skills or learn new information?

9 responses



Appendix D – Challenges & Issues Encountered

Challenges	Justification
Learning Flask	Having no prior experience programming with this Python framework. The first step was grasping the basics and familiarising myself with Flask, with the aid of several resources such as Flask documentation and online tutorials.
API Integration	After having grasped a solid foundation of Flask, the next step was to connect the backend with the React.js Frontend, which proved to be more difficult than expected. After having implemented both the frontend and backend an issue faced was when communication between the client and server wasn't taking place, which was later found out to be due to a Cross-Origin Resource Sharing (CORS) issue. To resolved this, the library 'flask-cors' was implemented to aid in handling fetch requests fetched and formatted from the frontend. In addition, utilising browser dev tools to debug requests and responses assisted in being able to integrate APIs.
Programming the Cognitive Games	A massive challenge in developing this application was creating the cognitive games that was functioning as intended. Several aspects had to be considered such as handling correct and incorrect responses as well as maintaining overall game flow. Additionally, games had to be clear, taking into account the primary stakeholder utilising this application. To aid with overall game flow, each of the games were separated into specific states (like 'showing', 'start' and 'result'). Moreover, all games shared certain functionalities such as timers, score tracking and submit and end buttons. These functionalities were converted into a React state components which was utilised across all games to maintain consistency.
Implementing Authentication	A vital part of this project was to create a secure Login and Signup features so that a user-specific experience can be achieved, so that users can track their progress on each of the games as well as view their highest scores. Alongside Flask being a completely new Python framework to work from, handling

	authentication and authorisation was another big and challenging step that had to be taken for this application to work as intended. The Python library 'bcrypt' was used to aid with password hashing and creating endpoints for login and signup in Flask. On the Frontend, the session tokens were stored in 'localStorage' and React was utilised to route users after having logged in.
Time Management	One huge challenge that was faced was timed management, several individual features had to be looked into such as the data handling in the backend, game implementation in the frontend, testing as well as focusing on other academic deadlines at university. Some issues with both the project as well as university deadlines took longer than expected. In order to tackle this, tasks were broken down into small sections and weekly goals were set in place to ensure deadlines were met.

Appendix E – Unit Testing Table

Tested Components	Test Case	Pass/Fail	Steps & Key Tools Utilised
LoginSignup.jsx	Ensuring visibility of the form input fields and buttons. Purpose: This is carried out to confirm that the input fields for 'username', 'email' and 'password' exist as well as the 'Signup' and 'Login' buttons.	Pass	React Testing Library (react-testing-library) Jest Steps: 1. Component is rendered inside a MemoryRouter 2. 'screen.getByPlaceholder()' was used to locate input fields for name, email and password 3. Buttons were located using 'screen.getByRole('button',)' 4. Checks visibility of inputs and buttons in this page.
LoginSignup.jsx	Ensuring that input fields can be filled out Purpose: That the component's states are updated accordingly as the user types	Pass	'fireEvent.change()' Steps: 1. Component is rendered using MemoryRouter 2. 'fireEvent.change()' is utilised to replicate user inputting name, email and password into input fields 3. '.value' is populated with correct input
LoginSignup.jsx	Ensuring that the signup form can be submitted with the input fields populated appropriately. Purpose: Axios POST call was mocked so that a real HTTP request doesn't take place. Testing that information is sent to '/signup'.	Pass	'axios.post.mockResolvedValueOnce' Steps: 1. 'axios.post' was mocked to simulate a POST call with correct information 2. Component is rendered 3. User inputs are filled in using 'fireEvent.change()' 4. Sign Up button is clicked using 'fireEvent.click()'

			5. Asserts that an axios POST call took place with the approporiate endpoint
LoginSignup.jsx	Ensuring that a test case is present if an email entered into the field is identical to one that's already in use. Purpose: Utilises axios to mock a failed signup and confirms that an alert takes place with an error output in the backend.	Pass	'act()' 'axios.post.mockRejectedValueOnce' Steps: 1. 'axois.post' was mocked to reject with appropriate message using 'axios.post.mockRejectedValueOnce()' 2. Window alert is also mocked using 'window.alert()' 3. Component is rendered, input fields are filled in and Sign Up button is clicked 4. Verify that 'window.alert' is called with correct backend error message
Login.jsx	Ensuring the visibility of form input fields and buttons. Purpose: This is carried out to confirm that the input fields for 'username', and 'password' exist as well as the 'Signup' and 'Login' buttons in the User Interface.	Pass	React Testing Library (react-testing-library) Jest Steps: 1. Component file is rendered in MemoryRouter 2. 'screengetByPlaceholderText()' is used to find input fields for name and password 3. Verify that login button is found using 'screen.getByRole()'
Login.jsx	Ensuring that input fields can be filled out Purpose: That the component's states for username and password are updated accordingly as the user types	Pass	'fireEvent.change()' Steps: 1. Component is rendered 2. Input fields are filled in using 'fireEvent.change()' 3. Check that input '.value' fields match the values inputted
Login.jsx	Ensuring that login form can be submitted and the data is sent to the API	Pass	'axios.post.mockResolvedValueOnce' Steps:

	Purpose: Utilises axios to mock a POST Request with the correct user information to '/login'.		1. 'axios.post' was mocked to simulate a successful post request with correct user information 2. Renders Component 3. Correct user information is inputted and Login button is clicked using 'fireEvent.click()' 4. Check that 'axios.post' was called with correct endpoint
Login.jsx	Ensuring that a test case is present if login is failed. Purpose: Utilises axios to mock a failed login and confirms that an alert takes place with an error output in the backend.	Pass	'axios.post.mockRejectedValueOnce' Steps: 1. Mock 'axios.post' to output an error message for invalid username and password 2. 'window.alert' is mocked in "window.alert = jest.fn();" 3. Component is rendered 4. Inputs are populated with incorrect information and Login button is clicked. 5. Verify that 'window.alert' was called with appropriate error message.
DigitSpan.jsx	Ensures that start screen is visible in UI Purpose: Ensures that both the 'Digit Span Test' title as well as the 'Start Game' button are visible on the screen when components are mounted	Pass	React Testing Library (react-testing-library) Jest Steps: 1. Component is rendered 2. Verify that Title and 'Start Game' is visible using 'screen.findByText' and 'screen.getByTest'
DigitSpan.jsx	Test to see if game transitions states from 'start' to 'showing' Purpose: To ensure that there is a sequence on screen after starting game	Pass	fireEvent screen.findBy() & screen.getByText() Steps: 1. Component is rendered

			2. 'Start Game' button is clicked using 'fireEvent.click()' 3. Verify that a sequence of numbers appear on screen by using 'screen.findByText()'
DigitSpan.jsx	Test to see if 'input' game state is present after transition from seeing the sequence with timer <u>Purpose:</u> Utilisation of fake timer to pass time and verification of appearance of input box after sequence display	Pass	'jest.useFakeTimers()' 'act()' screen.findBy() & screen.getByText() <u>Steps:</u> 1. Component is rendered 2. Since sequence display is shown for a duration of second, 'jest.useFakeTimers()' is utilised to fast forward the display. 3. Verify that input box appears after seeing the sequence display using 'screen.findByPlaceholderText()'.
DigitSpan.jsx	Test to see appearance of 'result' game state after inputting wrong answer <u>Purpose:</u> Inputting wrong answer into input field and verifying if 'Game Over' text is present on screen	Pass	'fireEvent.change()' screen.findByText() & screen.getByText() <u>Steps:</u> 1. Component is rendered 2. Start game and fast forward screen display 3. Input incorrect sequence using 'fireEvent.change()' 4. Check if 'Game Over' text is displayed using 'screen.findByText()'
DigitSpan.jsx	Test to see if scores and levels are incremented on correct input <u>Purpose:</u> Inputting correct answer into input field and verifying game progresses to next level	Pass	'jest.advanceTimersbyTime()' 'fireEvent.change()' screen.findBy() & screen.getByText() <u>Steps:</u> 1. Component is rendered

			2. Start game and fast forward screen display 3. Input correct sequence, by getting actual answer through 'screen.findByText()' 4. Press Submit button 5. Check if 'Remember this Sequence' txt appears on screen, indicating game has progressed due to correct answer
DigitSpan.jsx	Test to see if empty input will give go to 'result' state as expected <u>Purpose:</u> Checking that submitting empty input will traverse to 'results' game state	Pass	'fireEvent.change()' screen.findBy() & screen.getByText() <u>Steps:</u> 1. Component is rendered 2. Game is started and empty input is submitted into input field 3. Verify that 'Game Over' text appears after this using 'screen.findByText()' to convey that empty input was handled appropriately.
DigitSpan.jsx	Test for failure when trying to fetch the user's high score from the backend <u>Purpose:</u> Ensuring that if a failed fetch takes place, the components are still rendered and game page remains	Pass	'jest.spyOn().mockRejectedValueOnce()' <u>Steps:</u> 1. window fetch is mocked using 'jest.spyOn().mockRejectedValueOnce()' for rejection when fetching high score 2. Component is rendered 3. Verify that components are still rendered despite erroring fetching high scores
DigitSpan.jsx	Tests for failure in backend when score is submitted <u>Purpose:</u> Ensuring that if the submission of score fails, the game will still transition	Pass	'fetch.mockRejectedValueOnce()' <u>Steps:</u> 1. Mocked 'window.fetch' to output appropriate error message when it fails to save the scores 2. Renders component

	as expected and go to the 'results' page		3. Plays the game to get to result screen 4. Verify that result screen shows despite score failing
PairedAssociateLearning.jsx	Test to see if 'Start Game' button as well as difficulties were rendering <u>Purpose:</u> Ensuring that when navigating to the start page, the necessary components are displayed such as the Title and the difficulties: 'Easy', 'Medium' and 'Hard'.	Pass	React Testing Library (react-testing-library) Jest <u>Steps:</u> 1. Component is rendered inside MemoryRouter 2.Both 'screen.findbyText' and 'screen.getByText' are utilised to try and find the difficulty buttons and the title 3. Verify that the title and difficulties are visible
PairedAssociateLearning,jsx	Test to see if slideshow appears after selecting appropriate difficulty <u>Purpose:</u> Ensures that after user clicks on their preferred difficulty, slideshow of pairs appears on the screen	Pass	fireEvent <u>Steps:</u> 1. Render the 'PairedAssociateLearning' component 2. Select difficulty to start slideshow using 'fireEvent.click()' 3. Verify the visibility of the slideshow on the screen using 'screen.findByText()'
PairedAssociateLearning.jsx	Test to see if there is a transition from the slideshow state (where pairs are shown) to the quiz state (where user inputs their answer) <u>Purpose:</u> Ensures that after the slideshow of the pairs, the game will automatically transition to the quiz state where users can input their answers.	Pass	act() jest.useFakeTimers() <u>Steps:</u> 1. Render the component 2. Select difficulty to start the slideshow 3. Utilisation of both 'jest.useFakeTimers()' and 'act()' to fast forward time when presenting the slideshows 4. Verify the appearance of input box during the quiz state using 'screen.findByText()'

PairedAssociateLearning.jsx	Test to see that if the correct answer is inputted and submitted, the quiz will continue Purpose: Ensures that after inputting the correct word for the pair and submitting, the quiz progresses to the next question	Pass	fireEvent Steps: 1. Render the component 2. Fast forward through the slideshow to transition to quiz state using 'jest.useFakeTimers()' and 'act()' 3. Input the correct response pair word to the input field 4. Click the submit button using 'fireEvent.click() 5. Verify that next question in quiz state is shown using 'screen.findByText()'
PairedAssociateLearning.jsx	Test to see if there is a transition to the results page after inputting a wrong answer Purpose: Ensure that the game acts appropriately and transitions the user to the results page after an incorrect input	Pass	fireEvent Steps: 1. Render the component 2. Start the game using 'fireEvent.click()' on 'Easy' mode 3. Enter an incorrect answer 4. Click submit using 'fireEvent.click()' 5. Verify that current screen is the results screen utilising 'screen.findByText()'
PairedAssociateLearning.jsx	Test for failure in backend when getting the high score from the backend Purpose: Ensures that if there is a failure in the backend, the game will still function as expected	Pass	jest.spyOn(window, 'fetch').mockRejectedValueOnce() Steps: 1. Mock a fetch using 'jest.spyOn(window, 'fetch').mockRejectedValueOnce()' to give an error when fetching the user's high score 2. Render the component 3. Verify that the Start screen loads despite this using 'screen.findByText()'
PairedAssociateLearning.jsx	Test for failure in backend when score is submitted	Pass	jest.spyOn(window, 'fetch').mockRejectedValueOnce()

	Purpose: Ensures that in the cases of failure in submitting the final score, the user will transition to the results page.		Steps: 1. Mock a fetch using ‘ jest.spyOn(window, ‘fetch’).mockRejectedValueOnce() to output an error when submitting the score 2. Start and play through game to reach results page 3. Click submit button using ‘fireEvent.click()’ 4.Verify that the current page is the results page using ‘screen.findByText()’
PairedAssociateLearning.jsx	Test to see the submission of an empty input automatically transitions users to the results page Purpose: Ensures that the game will respond accordingly by ending the game, if there is an empty input submitted	Pass	fireEvent.change() Steps: 1. Render the component 2. Start the quiz and submit and empty input using ‘fireEvent.change()’ 3. Click submit button 4. Verify that the page transitioned to is the results page using ‘screen.findByText()’
PairedAssociateLearning.jsx	Test to see that answers, which are correct, are accepted even if they are in upper or lower case Purpose: Ensures that correct answers are accepted regardless of their case	Pass	fireEvent Steps: 1. Render the component 2. Start the quiz 3. Input correct answer in upper case using ‘.toUpperCase()’ 4. Verify that quiz progresses to next question using ‘screen.findByText()’
SelfOrderedSearch.jsx	Test to see if ending the game by clicking the ‘End Game’ button navigates the user to the results page Purpose: Ensures that by clicking the ‘End Game’ button, the component functions	Pass	React Testing Library (react-testing-library) Jest Steps:

	accordingly and transitions from the game page to the results page		1. Mock token is initialised in 'sessionStorage' to convey a user logged in 2. Render the component in MemoryRouter 3. Click Start Game button using 'fireEvent.click()' 4. After transitioning to the game page, click on 'End Game' button using 'fireEvent.click()' 5. Verify that the page navigated to is the results page using 'screen.getAllByText()'

Appendix F – Updated Functional & Non-Functional Requirements

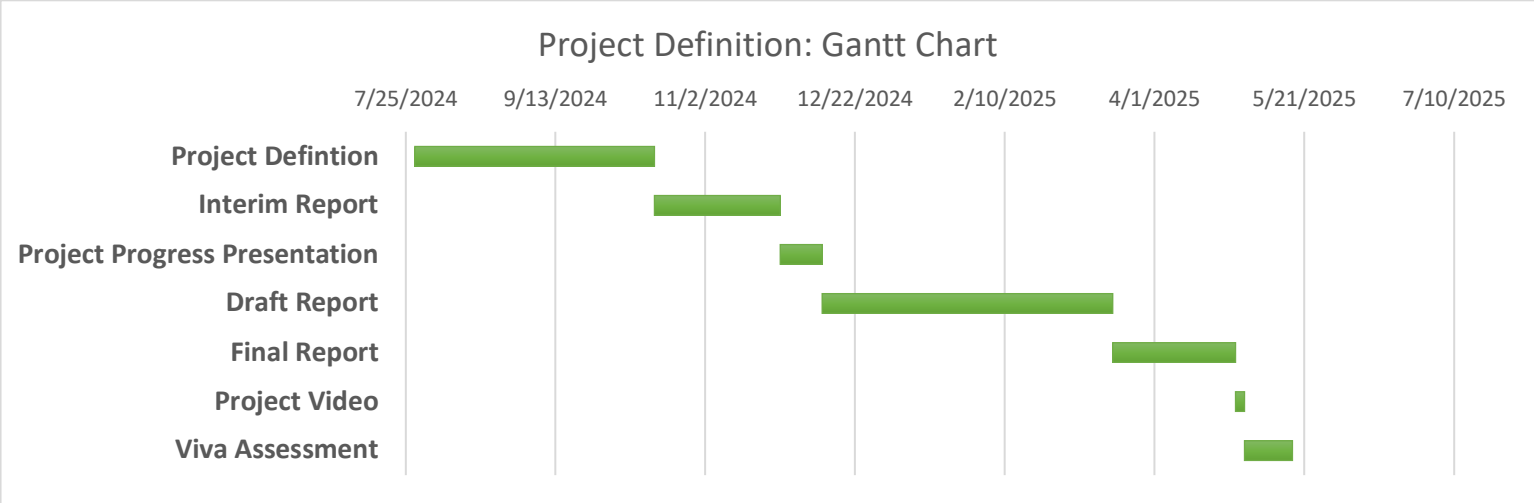
ID	Functional Requirements
F1	The system must be able to render the Signup form input fields: name, email and password as well as the 'Signup' and 'Login' buttons in the Login and Signup pages.
F2	The system must allow the user to input onto the fields and update the component's state variable.
F3	The system must handle the signup process successfully and send the data from the signup form to the appropriate backend endpoint '/signup'
F4	The system must display an error message if signup and login process fail due to duplicate email addresses or invalid credentials.
F5	The Digit Span Test must display a start screen as well as a 'Start Game' button to commence the games
F6	The Digit Span Test must display a sequence of numbers for a specific duration of time after the user starts the game
F7	The Digit Span Test must allow the user to progress to the next level if they input the correct answer (their input matches the sequence)
F8	The Digit Span Test must be able to handle incorrect answers by transitioning the user to the results page
F9	The Digit Span Test must pass the user's highest and most recent score to the backend
F10	The Digit Span Test must display the user's highest and most recent score in the results page
F11	The Paired Associate Learning game must display a start screen to display the game name as well as instructions
F12	The Paired Associate Learning game must allow users to select their preferred difficulty
F13	The Paired Associate Learning game must display a slideshow of word pairs for the user to see after they selected their difficulty

F14	The Paired Associate Learning game must accept the answer from the user during the quiz state, regardless of whether the answer is upper or lower case
F15	The Paired Associate Learning game must be able to handle incorrect answers appropriately by traversing the user to the results page
F16	The Paired Associate Learning game must be able to submit the highest and most recent score from the user to the backend
F17	The Paired Associate Learning must display the user's highest and most recent score in the results page
F16	The Self Ordered Search must display a start screen with a 'Start Game' button
F17	The Self Ordered Search must be able to traverse through different game states, depending on the actions carried out by the user
F18	The Self Ordered Search must be able to send the highest and most recent score of the user to the backend
F19	The Self Ordered Search must display the user's highest and most recent score in the results page

ID	**Non-Functional Requirements**
NF1	The web application must be able to maintain user login sessions through the use of the user's session token
NF2	Error feedback from the application must take place and be displayed at least 2 seconds after the user carries out an invalid action
NF3	There must be a smooth transition through game states (e.g. start state, showing state, result state) in all games in the application
NF4	The system must not break even if an error occurs in the backend, such as a failed backend score submission
NF5	The application must be simplistic to be more tailored toward the target audience
NF6	The games in the application must operate without page refreshes

NF7	The code must adhere to PEP8 code format standards for Python to maintain consistency
NF8	The system must put emphasis on the use of strong passwords in the Login and Signup pages
NF9	The System must be able to handle multiple users and data being inputted

Appendix G – Initial Development Stage: Gantt Chart



Additional Appendices (as needed)