

Інститут інформатики і радіоелектроніки,
Факультет комп'ютерних наук та технологій
(повне найменування інституту, назва факультету)

Кафедра системного аналізу та обчислювальної математики
(повна назва кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

бакалавр

(рівень вищої освіти (освітньо-кваліфікаційний рівень))

на тему Розробка програмного забезпечення для тренування
дітей з розсіяною увагою

Виконав: студент 4 курсу, групи КНТ-818сп
напряму підготовки (спеціальності)

124 – Системний аналіз

(код і назва напряму підготовки, спеціальності)

Ворона В. Р.

(прізвище та ініціали)

Керівник

Рябененко А. Є.

(прізвище та ініціали)

Рецензент

Пирожок А. В.

(прізвище та ініціали)

Запоріжжя - 2021 року

Національний університет «Запорізька політехніка»
(повне найменування вищого навчального закладу)

Інститут, факультет _____
Кафедра _____
Рівень вищої освіти (освітньо-кваліфікаційний рівень) _____
Напрямок підготовки 8.040303-Системний аналіз і управління
Спеціальність 124-Системний аналіз
(код і назва) (код і назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри _____

“ 8 ” березня 20__ року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТУ

Ворона Вадим Романович

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Розробка програмного забезпечення для тренування дітей з розсіяною увагою

керівник проекту (роботи) Рабенко А.Є доцент кафедри к.ф.-м.н.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від “__” ____ 20__ р. №__

2. Строк подання студентом проекту (роботи) _____
3. Вихідні дані до проекту (роботи) створено застосування «Play Attention».
Розроблюваний програмний продукт призначений для підвищення уваги,
поведінки і когнітивних функцій у дітей хворих на СДУГ.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області, 2. Вимоги до розроблюваного програмного продукту, 3. Основні рішення щодо реалізації системи, 4. Керівництво програміста, 5. Керівництво користувача

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Слайди презентації

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	Рибенко А.Є. доцент кафедри ІТФ-М.Ч.		
Нормоконтроль	Широкозад Д.В.		

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Аналіз предметної області	1 тиждень	
2	Розробка та затвердження технічного завдання	2 тиждень	Технічне завдання
3	Проектування структури системи	3-4 тиждень	Структурна схема
4	Розробка схеми функціонування системи та модулів	5-6 тиждень	Функціональна схема Модулі
5	Створення програмного коду системи	7-8 тиждень	Текст програми
6	Тестування та відлагодження системи	9 тиждень	Працевздатність системи
7	Розробка розділів ПЗ	10 тиждень	ПЗ
8	Розробка слайдів презентації	11-12 тиждень	Слайди презентації
9	Захист випускної роботи	13 тиждень	

Студент

(підпис)

(прізвище та ініціали)

Керівник проекту (роботи)

(підпис)

(прізвище та ініціали)

РЕФЕРАТ

ПЗ: 74 с., 23 рис., 0 табл., 3 дод., 4 джерел.

Об'єкт дослідження – методи психологічного впливу на дітей з синдромом дефіциту уваги і гіперактивності.

Мета роботи – створення програмного забезпечення для підвищення уваги, поведінки і когнітивних функцій у дітей та дорослих з синдромом дефіциту уваги і гіперактивності.

Для досягнення поставленої мети необхідно виконати наступні задачі:

- дослідити предметну область;
- виконати аналіз процесу розробки;
- розробити структуру застосунка;
- виконати програмну реалізацію застосунка;
- оформити пакет документації на створений програмний продукт та

провести його тестування.

Програмний продукт, який створено в результаті виконання випускної кваліфікаційної роботи, дозволяє підвищити увагу, поведінку і когнітивні функції у дітей та дорослих з СДУГ.

Програмне забезпечення створено з використанням технології NW.js, сумісне з x86- та x64-бітними платформами на базі ОС Windows, Mac OS X та Linux. Система має зручний, максимально орієнтований на користувача інтерфейс.

PLAY ATTENTION, СДУГ, УВАГА, ПОВЕДІНКИ, КОГНІТИВНІ ФУНКЦІЇ, NW.JS, ANGULARJS, LOKIJS

ЗМІСТ

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ	2
РЕФЕРАТ	4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ ТА ТЕРМІНІВ	7
ВСТУП.....	8
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ.....	10
1.1 Заходи щодо вирішення проблеми СДУГ	10
1.2 Огляд існуючих програмних застосувань	10
2 ВИМОГИ ДО РОЗРОБЛЮВАНОВОГО ПРОГРАМНОГО ПРОДУКТУ	12
2.1 Загальне уявлення	12
2.2 Студент.....	12
2.3 Тренер.....	13
2.4 Поведінка	13
2.5 Нагороди	14
2.6 Бали.....	14
2.7 Система 5 зірок.....	14
2.8 Об'єктивна поведінка	15
2.9 Sheer Genius	15
2.9.1 Персонажі.....	16
2.9.2 Реалізація	16
2.10 Ролі	16
2.10.1 Студент.....	16
2.10.2 Тренер.....	17
2.10.3 Адміністратор.....	17
2.11 Приладова панель.....	17
3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ СИСТЕМИ.....	18
3.1 Мова програмування.....	18

3.2	Вибір інструментальних засобів.....	19
3.3	СУБД	19
3.4	Архітектура.....	20
3.5	Дизайн	22
3.6	Структура проекту	26
3.6.1	Контролери	28
3.6.2	Сервіси	28
4	КЕРІВНИЦТВО ПРОГРАМІСТА	30
4.1	Підняття проекту.....	30
4.1.1	Встановлення необхідних пакетів.....	30
4.1.2	Клонування та залежності.....	31
4.1.3	Складання та запуск.....	31
4.2	Відладка	31
4.3	Опис функцій.....	32
4.4	Подальша розробка проекту	32
5	КЕРІВНИЦТВО КОРИСТУВАЧА	33
5.1	Призначення та умови використання програми	33
5.2	Виконання програми.....	33
5.3	Використання програми	35
5.3.1	Адміністратор.....	35
5.3.2	Тренер.....	38
5.3.3	Студент.....	41
5.4	Повідомлення оператору.....	43
	ВИСНОВКИ.....	44
	ПЕРЕЛІК ПОСИЛАНЬ	45
	Додаток А Технічне завдання	46
	Додаток Б Текст програми	49
	Додаток В Слайди презентації.....	68

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ ТА ТЕРМІНІВ

ОП – оперативна пам'ять;

ОС – операційна система;

ПК – персональний комп'ютер;

ПЗ – програмне забезпечення;

Синдром дефіциту уваги і гіперактивності (СДУГ) - неврологічно-поведінковий розлад розвитку, що починається в дитячому віці;

СУБД – система управління базами даних;

ТЗ – технічне завдання;

AngularJS – JavaScript-фреймворк з відкритим програмним кодом призначений для розробки односторінкових застосунків;

DOM (Document Object Model) – специфікація прикладного програмного інтерфейсу для роботи зі структурованими документами;

Hardware Abstraction Layer (HAL) – демон, що надає шар апаратних абстракцій;

JS (JavaScript) – динамічна, об'єктно-орієнтована мова програмування;

Lodash – сучасна JavaScript бібліотека утиліт;

MVC (Model-View-Controller) – архітектурний шаблон, який використовується під час проектування та розробки програмного забезпечення;

Node.js – платформа з відкритим кодом для виконання високопродуктивних мережевих застосунків, написаних мовою JavaScript;

NW.js – дозволяє викликати всі модулі Node.js безпосередньо з DOM і створення програмного забезпечення з усіма веб-технологіями;

Play Attention (PA) – назва застосування;

TS (TypeScript) – мова програмування, що розширює можливості JavaScript.

ВСТУП

Синдром дефіциту уваги і гіперактивності — неврологічно-поведінковий розлад розвитку, що починається в дитячому віці [1]. Проявляється такими симптомами, як труднощі концентрації уваги, гіперактивність і погано керована імпульсивність. В існуванні СДУГ сумніваються низка медиків, вчителів, політиків, батьків і засобів масової інформації. Одні вважають, що СДУГ не існує взагалі, інші вірять, що є генетичні та фізіологічні причини цього стану. Але проблема існує і повсякчас. Ціла низка дітей, молоді і навіть дорослих страждають від СДУГ.

Головна проблема синдрому дефіциту уваги і гіперактивності полягає в тому, що й досі в науково-медичному світі не було прийнято остаточне рішення щодо доцільного лікування цієї хвороби. На сьогоднішній день існує достатня кількість методик, котрі обіцяють лікувати хворих на СДУГ за рахунок відповідних ліків; контролювати їхню поведінку і симптоматику та поліпшити загальний стан перебігу СДУГ за допомогою застосунків. Є багато «пізнавальних ігор», які говорять, що вони зроблять ваш мозок краще через «нейропластику». Але більшість цих проектів не досягають поставленої від самого початку мети: загалом вони нагадують прості записники, які допомагають систематизувати щоденні справи і представити їх у наглядному вигляді. Тож, проблема цілком не вирішується, а ідеї таких застосунків стають все більш безглуздими і не практичними.

В даній роботі розглядається застосунок під назвою «Play Attention» [2]. Це – освітня система для підвищення уваги, поведінки і когнітивних функцій. Play Attention має нарукавну пов'язку, яка вимірює активність мозку на увагу. Суть полягає в тому, що дитина може активувати наші пізнавальні ігри, застосовуючи всю свою увагу, щоб розпочати гру. Потім, використовуючи цю ж технологію, вона дійсно може рухати ігрових персонажів за допомогою розуму/уваги. Якщо вона втрачає увагу, гра зупиняється, поки вона повністю не

застосує себе знову. Таким чином пізнавальні ігри + Play Attention монітор уваги = величезні відмінності в успіху.

Проект побудований на базі сучасних технологій, таких як NW.js, що базується на технологіях Node.js та WebKit, а також AngularJS, який відповідає за front-end застосунка.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Заходи щодо вирішення проблеми СДУГ

Синдром дефіциту уваги і гіперактивності – неврологічно-поведінковий розлад розвитку, що починається в дитячому віці. Проявляється такими симптомами, як труднощі концентрації уваги, гіперактивність і погано керована імпульсивність.

Немає жодного відомого лікування цієї хвороби, але є цілий ряд методів, які можуть допомогти з управлінням симптомів. Процедури варіюються від впливу на поведінку до ліків за рецептом. У більшості випадків ліки є ефективним методом лікування СДУГ. Проте Національний інститут психічного здоров'я передбачає, що інші методи теж дуже важливі. До таких методів відносяться: психотерапія, поведінкова терапія, тренування соціальних навичок, групи підтримки, виховання.

В наш час інформаційних технологій можна створити таке програмне забезпечення, яке змогло б допомогти в лікуванні СДУГ. Аби досягти цього, нами було прийнято рішення створити відповідний застосунок, котрий може фіксувати відхилення уваги дитини у реальному часі під час проходження нею гри. Це повинно допомогти лікарю, вчителю або батькам слідкувати за увагою, поведінкою дитини, і в той же час бути цікавим самій дитині.

1.2 Огляд існуючих програмних застосувань

Наразі на ринку є дуже багато «пізнавальних ігор», які обіцяють покращити функціонування мозку за допомогою нейропластичності, або застосунків, які допомагають в організації робочого процесу. Але у цих застосунків є багато мінусів та їхня ефективність остається під сумнівом. Жодний з цих застосунків не надає комплексного набору інструментів, щоб

допомогти дитині та батькам впоратися з хворобою. Більшість застосунків має лише розважальний контент, який не містить жодного практичного навантаження. Їхня сутність полягає у швидкості проходження хворого на СДУГ гри, але кількість помилок за гру не є показником і прогресом в лікуванні дитини – це лише суха статистика.

Отже, розробка застосунка для допомоги дітям із СДУГ є релевантним в наш час, тому що немає жодного застосунка, який би повністю був направлений на вирішення даної проблеми.

2 ВИМОГИ ДО РОЗРОБЛЮВАНОВОГО ПРОГРАМНОГО ПРОДУКТУ

2.1 Загальне уявлення

Розроблюваний програмний продукт призначений для допомоги людям хворим на СДУГ поліпшити їх здатність ефективно витратити увагу.

Система будується навколо відносин «тренер-студент». Студент – користувач, який потребує поліпшення здатності ефективно витратити увагу. Використовуючи Play Attention, студент робить це граючи в ігри під час носіння пристрою BodyWave. Пристрій BodyWave забезпечує зворотній зв'язок з застосунком в режимі реального часу, фіксує увагу студента і впливає на гру. Елементи управління на екрані визначають, коли студент може взаємодіяти з застосунком і впливати на процес гри. Тренер допомагає студенту з використанням системи і обговорює з ним (або з представником хворого на СДУГ) його прогрес.

2.2 Студент

Студенти грають сеанси. Сеанс являє собою набір ігор в які студент буде грати за один раз. Сеанс починається з огляду попереднього сеансу, потім дозволяє студенту грати в ігри. Завершуючи сеанс, студент має залишити певну замітку щодо пройдених ним ігор. У той час, як студент грає в ігри кожного щотижневого сеансу, він може заробити бали в залежності від його продуктивності в іграх. Студент заробляє бали під час досягання цілей, встановлених системою. Бали можуть бути використані для отримання нагород, створених тренером. Нагороди створюються для кожного студента на базі його позитивних стимулів. Наприклад це може бути морозиво на десерт, додаткові

30 хвилин телевізора, білети у кіно. Це лише буденні дрібниці, але вони можуть стати справжньою мотивацією для хворого на СДУГ.

2.3 Тренер

Тренер допомагає направляти студента в процесі навчання і стежити за поведінкою студента в той час, як він грає в ігри. Після кожної гри тренер оцінює певну поведінку, і таким чином вона може бути відслідкована системою. Тренер відповідає за створення щотижневого розкладу сеансів, а потім проводить кожен з них. Тренер також створює нагороди. Студент може купити нагороди використовуючи бали, які він заробив за досягнення цілей. Додатково тренер може створювати звіти про продуктивність студента і відправити їх до «Кола успіху» студента, якщо це можливо. Коло успіху – це список людей, які підтримують студента і надають підтримку за межами системи Play Attention.

2.4 Поведінка

Одним з основних аспектів системи є «поведінка». У той час як тренер дивиться за студентами, котрі проходять сеанс, вони будуть демонструвати схожі фізичні моделі поведінки, які відбуваються під час кожної гри. Вони часто згадуються як «самостійно відволікаючі» або «самостійно стимулюючі» поведінки. В кінці кожної гри, тренер для відстеження кожної поведінки буде ставити оцінку в системі. Якщо, наприклад, студент втратив увагу 5 разів під час конкретної гри, тренер поставе цифру 5 біля поведінки «Непосида» одноразово після гри. Частиною огляду сеансу є графік поведінки, тому студент чітко побачить що він робить і як часто. Тренер обирає одну поведінку на

сеанс, яка має назву «Об'єктивна поведінка». Метою цього є зниження прояву тієї чи іншої симптоматики певної поведінки.

2.5 Нагороди

Нагороди отримуються завдяки заробленим балам. Це можуть бути будь-які нагороди, які тренер вважатиме за мотивацію для студента. Наприклад: «15 хвилин iPad», «піца-вечірка з друзями», «прогулянка парком протягом дня» або «білети у кіно». Тренер самостійно обирає потрібні нагороди, а потім встановлює їх вартість (бали, які необхідні для їхнього отримання). Студент може витратити свої зароблені бали на різні винагороди. Винагороду можна «купити і використати» або «купити і використати пізніше».

2.6 Бали

Бали нараховуються після досягнення цілей. 1 бал за досягнення ігрової цілі і 1 бал за ціль «Об'єктивна поведінка» на сеанс. Студенти матимуть можливість витратити свої бали в розділі «Нагороди» своєї приладової панелі. Бали завжди відображаються на приладовій панелі студента в правому верхньому кутку.

2.7 Система 5 зірок

Для того, щоб студент перейшов від початкового до середнього, або від середнього до продвинутого рівня гри, потрібно задовольнити певні критерії. Ці критерії різні для кожної гри, але здебільшого, головним критерієм є увага, яка повинна бути 75% або вище. Якщо студент може зробити це 5 разів поспіль,

програма спонукає його перейти до наступного рівня. Таким чином, кожна зірка є спробою в межах індивідуальної гри.

2.8 Об'єктивна поведінка

Об'єктивна поведінка – це ціль на сеанс, а не на одну гру. Зазвичай він вибирається з основних поведінок студента тренером.

2.9 Sheer Genius

Цілі на сеанс встановлює «Sheer Genius». Sheer Genius – це набір персонажів і штучний інтелект, який допомагає вибрати ігри в які студент повинен грати в своїх сеансах і цілі, які студент має досягти для отримання балів під час гри. SG також виступає в якості персонажа для допомоги студенту під час його сеансів (рис. 2.1).



Рисунок 2.1 – Персонаж Sheer Genius

2.9.1 Персонажі

Усього буде 6 персонажів. Кожна вікова група буде мати 2 персонажа, одна жінка і один чоловік.

Цей персонаж буде знаходитись, як правило, у лівій стороні під панелью навігації і матиме текстове поле в якому буде показано цілі на поточний сеанс або вітання, коли студент досягає успіху.

2.9.2 Реалізація

Це штучний інтелект для пропонованих ігор під час створення сеансу тренером, цілей, а також нагородження балами. Sheer Genius також буде визначати, коли студент переходить на наступний рівень гри.

Перші 7 сеансів система збирає базову статистику по продуктивності студента. Після перших сеансів, Sheer Genius запропонує ігри базуючись на продуктивності студента в іграх. SG також створить цілі для кожної гри після перших 7 результатів.

2.10 Ролі

Роль – група зумовлених дозволів. Програмний продукт буде включати в себе 3 ролі: Студент, Тренер, Адміністратор.

2.10.1 Студент

Це користувач, який грає в ігри, щоб поліпшити симптоматику певної поведінки.

Дозволи:

- грати в ігри.

2.10.2 Тренер

Це користувач, який буде проводити навчання студента.

Дозволи:

- грати в ігри без моніторингу статистики;
- управління сеансами;
- управління студентами.

2.10.3 Адміністратор

Це користувач, який займається налаштуванням встановленого застосунка.

- грати в ігри без моніторингу статистики;
- управління студентами і тренерами;
- менеджер апаратних засобів;
- управління системними налаштуваннями.

2.11 Приладова панель

У кожного типа користувача буде своя приладова панель зі своїм набором функцій.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ СИСТЕМИ

3.1 Мова програмування

Мовою програмування є TypeScript та JavaScript ES6. JavaScript та TypeScript [3] є одними з найпоширеніших мов, що використовуються у сфері веб-розробок в наш час. Раніше JavaScript використовувався лише на клієнтській частині, проте тепер це один із найпопулярніших мов програмування, яку використовують як на клієнтській, так і на серверній частині. А TypeScript розширює можливості Javascript та додає до нього структуру ООП.

Головні переваги JavaScript:

- кросплатформенність: майже весь JavaScript-код можна без змін використовувати на комп'ютері, що працюють під управлінням різних операційних систем;
- легкий порт застосунка для мобільних платформ. JS застосунки можуть бути використані на Android/iOS за допомогою проекту PhoneGap;
- JS застосунки мають добре налагоджені технології для тестування (Selenium для інтеграційного тестування, Karma + Jasmine для модульного тестування);
- використання реактивних фреймворків – AngularJS, дозволить зберегти дуже ясний поділ між моделью та видом;
- дуже велика швидкість розробки;
- можна використовувати JS-консоль для експериментів під час виконання застосунка;
- гнучкість;
- підтримка майже всіх типів баз даних;
- наявність вихідного коду та безкоштовність (стратегія Open Source).

3.2 Вибір інструментальних засобів

Для проекту була обрана зв'язка NW.js + AngularJS.

Так як застосунок повинен бути кросплатформним було вирішено взяти NW.js [4]. NW.js (node-webkit) – дозволяє викликати всі модулі Node.js безпосередньо з DOM і створювати програмне забезпечення з усіма веб-технологіями. Це дало нам можливість застосувати AngularJS [5] для Front-End. AngularJS – це JavaScript-фреймворк призначений для розробки односторінкових застосунків. І усі Node.js модулі для Back-End. За рендер застосунка відповідає WebKit.

3.3 СУБД

СУБД обрано LokiJS. LokiJS [6] – це документо-орієнтована JavaScript база даних, дещо схожа на MongoDB.

Переваги LokiJS:

- NoSQL з синхронізацією та персистентністю;
- Redis-стиль, in-memory база даних, дає можливість моментального доступу до даних;
- Її легко встановити за допомогою пакетного менеджера npm.

База даних містить наступні колекції:

- users, дана колекція зберігає дані про користувачів (студенти, тренера, адміністратори);
- games, дана колекція зберігає дані про ігри;
- banks, дана колекція зберігає дані про зароблені бали;
- rewards, дана колекція зберігає дані про нагороди;
- sessions, дана колекція зберігає дані про сесії;

- wantedRewards, дана колекція зберігає дані про бажані нагороди студентів;
- studentRewards, дана колекція зберігає дані про куплені нагороди студентами;
- gameResults, дана колекція зберігає дані результатів ігор студентів;
- goals, дана колекція зберігає дані ігрових цілей студентів;
- systemNotifications, дана колекція зберігає дані про системні повідомлення, які необхідно показати користувачам у різних ситуаціях.

3.4 Архітектура

Архітектура Play Attention (рис. 3.1) буде складатися з наступних модулів:

- Застосунок (NW.js + AngularJS) – відповідає за більшу частину загальної логіки;
- Ігри – написані на мові програмування Java та ігровому движку jMonkeyEngine;
- Ігрові додатки – написані на мові програмування Python;
- Hardware Abstraction Layer – демон, що надає шар апаратних абстракцій.

Архітектурним шаблоном застосунка буде MVC.

Застосунок складається з двох головних частин:

- «Незалежний пристрій» – відповідальний за UI, є взаємодія з користувачем (в більшості сценаріїв використання студентів і тренерів);
- «Залежний пристрій» – відповідає за запуск та взаємодію між зовнішніми процесами (ігри, HAL).

Підмодуль «Незалежний пристрій»

Відповідальний за UI, взаємодіє з користувачем, іграми і пристроями.
Написано на AngularJS + TypeScript.

Підмодуль «Залежний пристрій»

Відповідальний за обмін між іграми, взаємодіє між зовнішніми процесами за допомогою Pipe та Protobuf.

Складається з 3-х частин:

- персистентність бази даних – відповідає за збереження бази даних;
- комунікатор – шина подій, яка дозволяє взаємодіяти між кожним процесом;
- менеджер процесів – відповідає за стар/стоп зовнішніх процесів.

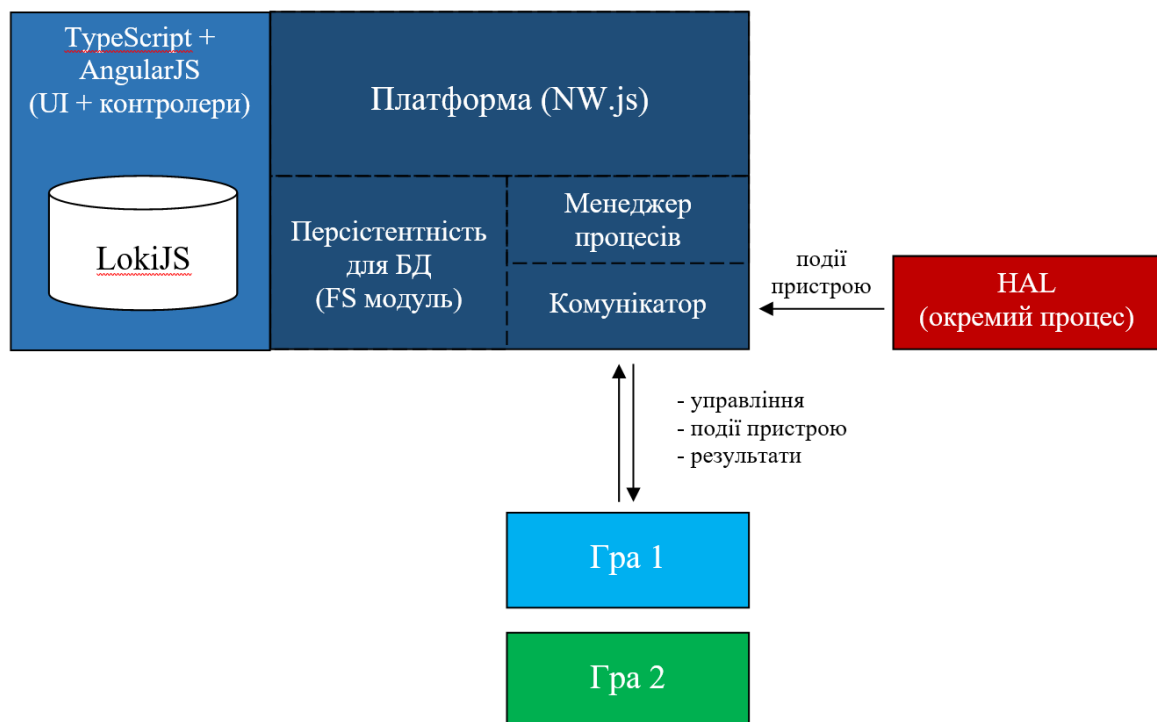


Рисунок 3.1 – Архітектура застосунка

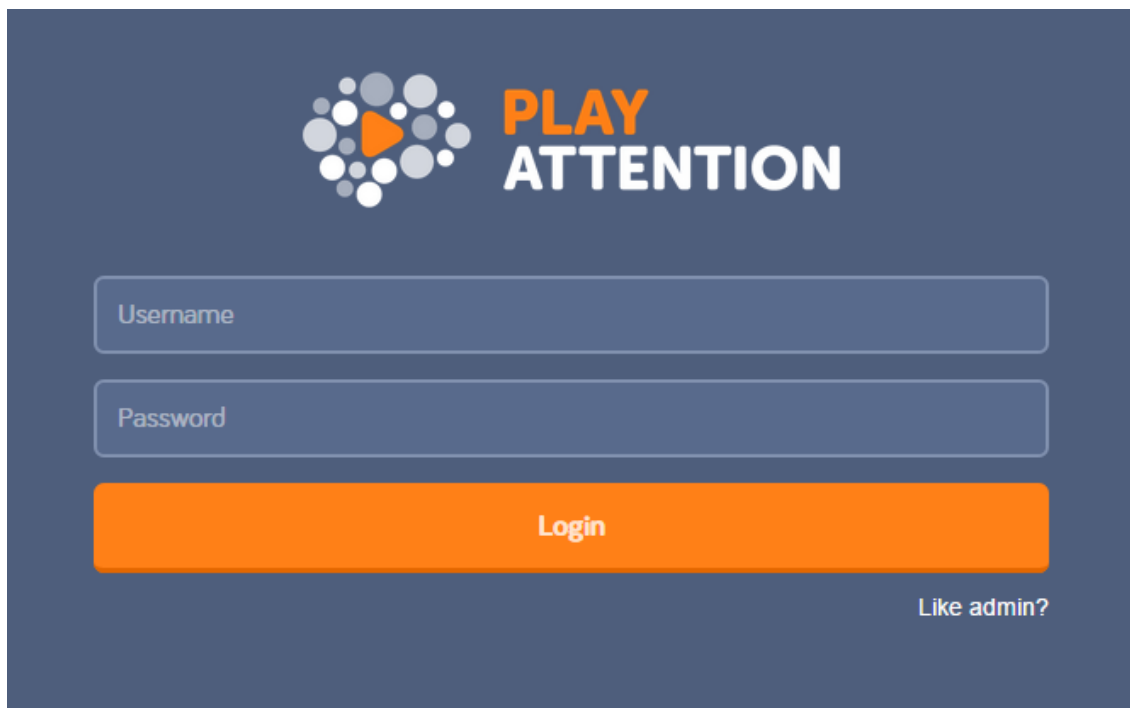
3.5 Дизайн

За основу для дизайну було вирішено взяти 3 кольори: синій, помаранчевий та білий. На базі цих кольорів був розроблений логотип проекту, який зображено на рисунку 3.2.



Рисунок 3.2 – Логотип проекту

На рисунку 3.3 зображена сторінка входу до приладової панелі різних користувачів (студент, тренер, адміністратор).



Username

Password

Login

Like admin?

Рисунок 3.3 – Вхід до приладової панелі

На рисунку 3.4 можна побачити «Огляд» сторінку студента на якій можна подивитися на результати попереднього сеансу.

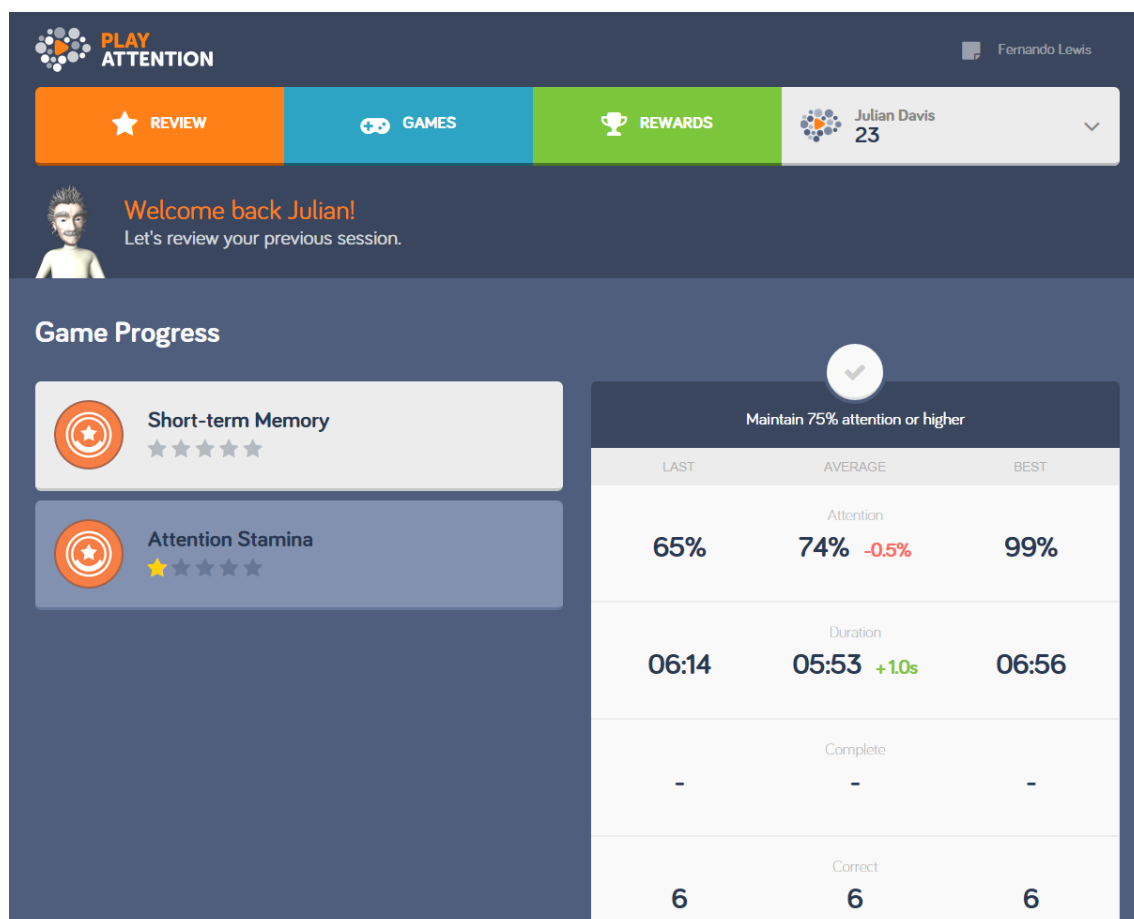


Рисунок 3.4 – «Огляд» сторінка студента

У лівій верхній частині сторінки знаходиться логотип проекту, у правій частині кнопка з іменем тренера при натисканні на котру можна написати замітку тренеру. Трохи нижче знаходиться головна навігація по приладовій панелі студента. Вона включає в себе усього декілька пунктів: «Огляд», «Ігри», «Нагороди» та випадаюче меню з фотографією студента, його ім'ям та заробленими балами. Випадаюче меню має 2 пункти: Профіль студента, де можна змінити фото, адрес електронної пошти та ім'я студента та вихід із сеансу. Нижче головної навігації знаходяться повідомлення від Sheer Genius. Під повідомленням знаходиться основний контейнер сторінки.

Студент може бути 3-х вікових груп: дитина, підліток, дорослий. Для кожної групи використовується своя цвітова схема для приладової панелі.

На рисунку 3.5 зображена сторінка розкладу сеансів приладової панелі тренера.

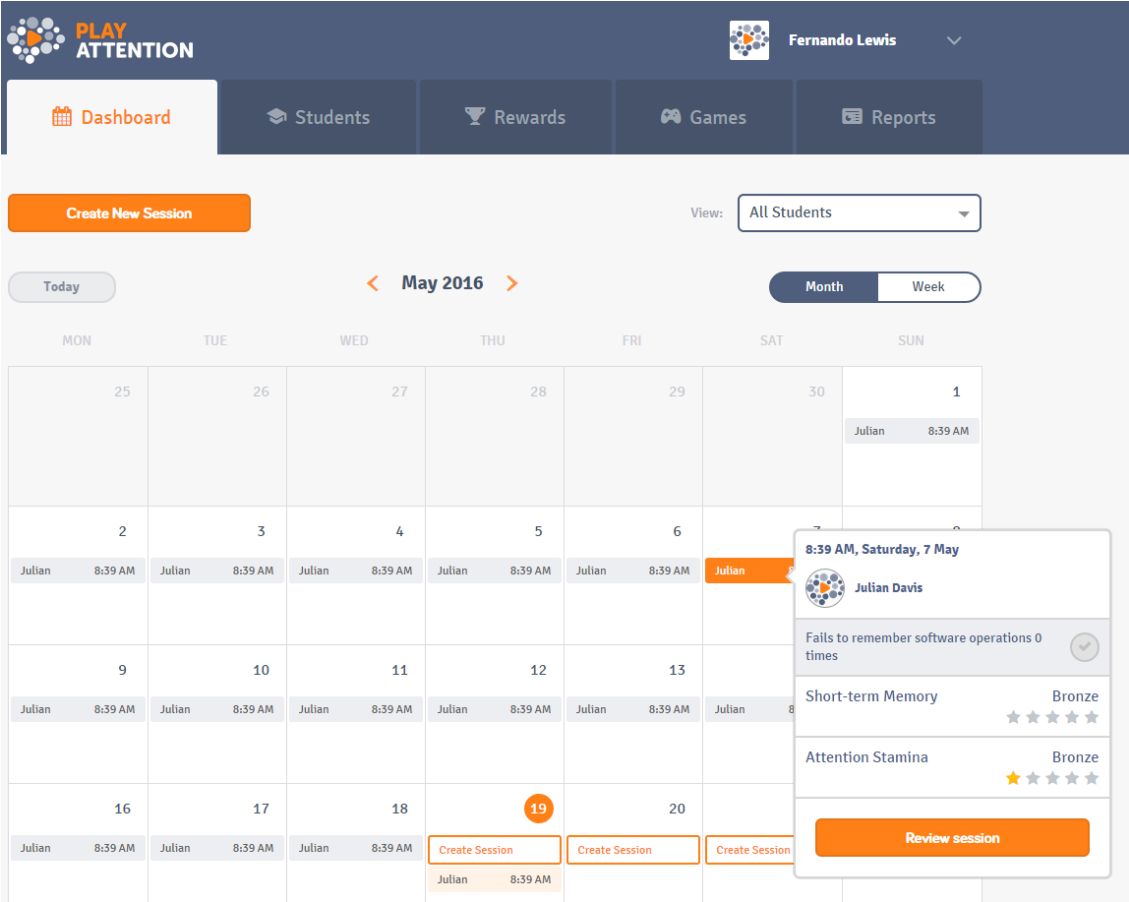


Рисунок 3.5 – Розклад сеансів

Кожний елемент повинен бути інтуєтивно зрозумілим, тому кожен елемент виглядає максимально просто. Таби навігації мають іконку поруч з написом, яка відображає суть сторінки. Більшість кнопок змінюють свій стан при взаємодії з ними та мають впливаючі підказки рис. 3.6.

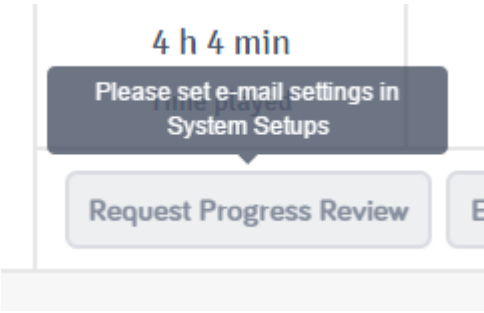


Рисунок 3.6 – Впливаюча пидказка

Приладова панель адміністратора не відрізняється дизайном від панелі тренера рис. 3.7.

PLAY ATTENTION System Administra... ▾

Dashboard Users Games Reports

System setup

Interface
Current language: English

English ▾

E-mail settings

Email authentication data

This Email will be used for functions such as **Request Progress Review** and **Email circle of success**. Make sure that the settings of this Email are enabled authentication with an unknown devices.

Application uses SMTP protocol for sending Emails

E-mail for sending

E-mail address
Email

Password Show
Password

E-mail service
E-mail service ▾

Secondary e-mail for Request Progress Review
Email

Save

Рисунок 3.7 – Сторінка системних налаштувань застосунка приладової панелі адміністратора

Перш ніж розпочати верстати сторінки по макетам було розроблено сторінку з усіма компонентами рис. 3.8. Це допомогло розробити набір базових CSS класів без дублювання та з коментарями. А також просту HTML-верстку.

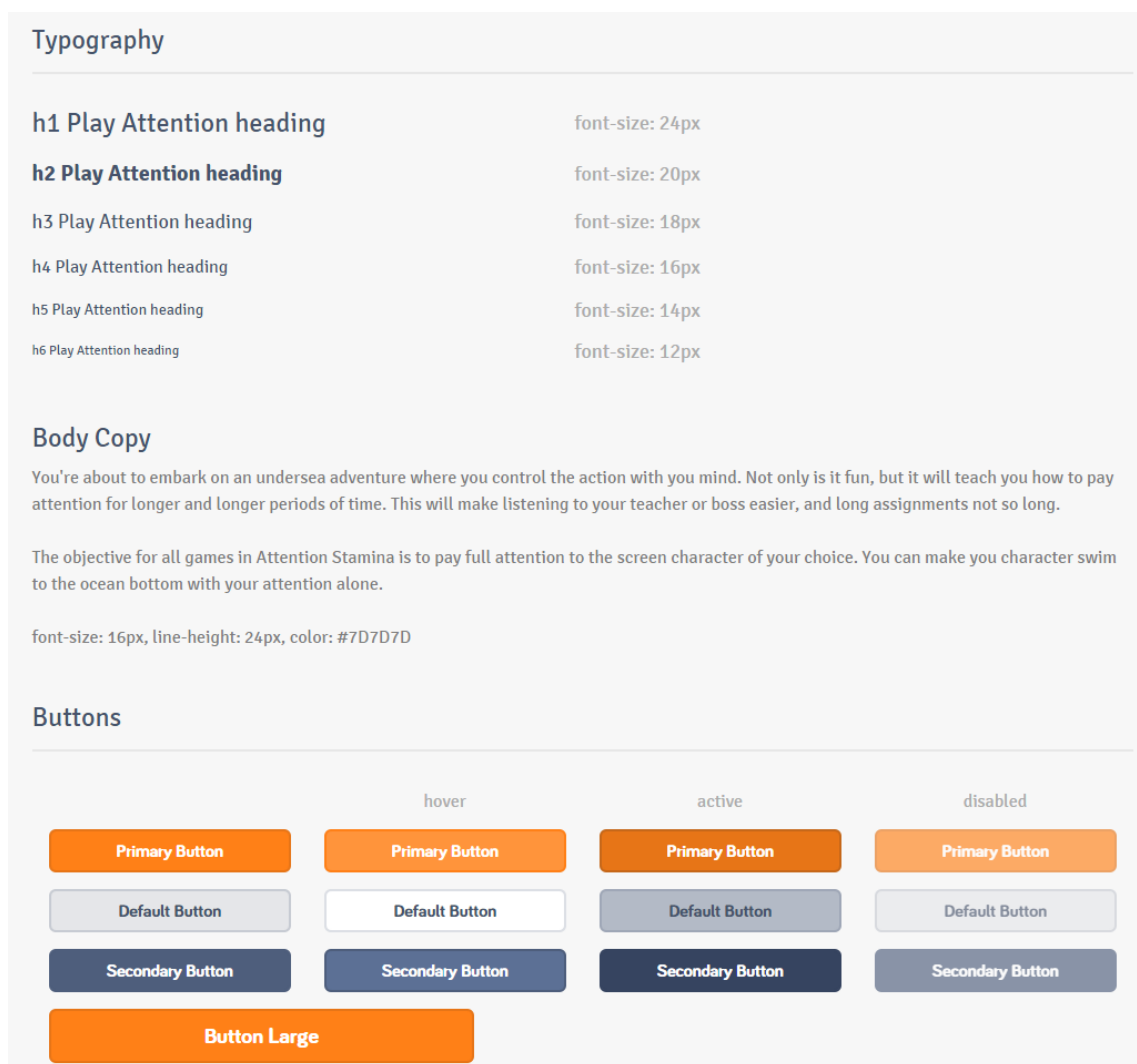


Рисунок 3.8 – Сторінка усіх компонентів застосунка

3.6 Структура проекта

Проект базується на платформі NW.js. Основний файл конфігурації NW-вікна застосунка та весь список необхідних node-модулей знаходиться у файлі `package.json`, який лежить у корні проекту. Поряд знаходиться база даних під назвою `play-attention.json` та конфігураційний файл застосунка `config.json` в якому знаходяться базові налаштування такі як: розклад бекапів, мова застосунка, знайдені девайси та інші.

Структура проекту базується на компонентах AngularJS таких як: контролери, директиви, фабрики, фільтри, сервіси та інші.

На рис. 3.9 можно побачити структуру проекту.

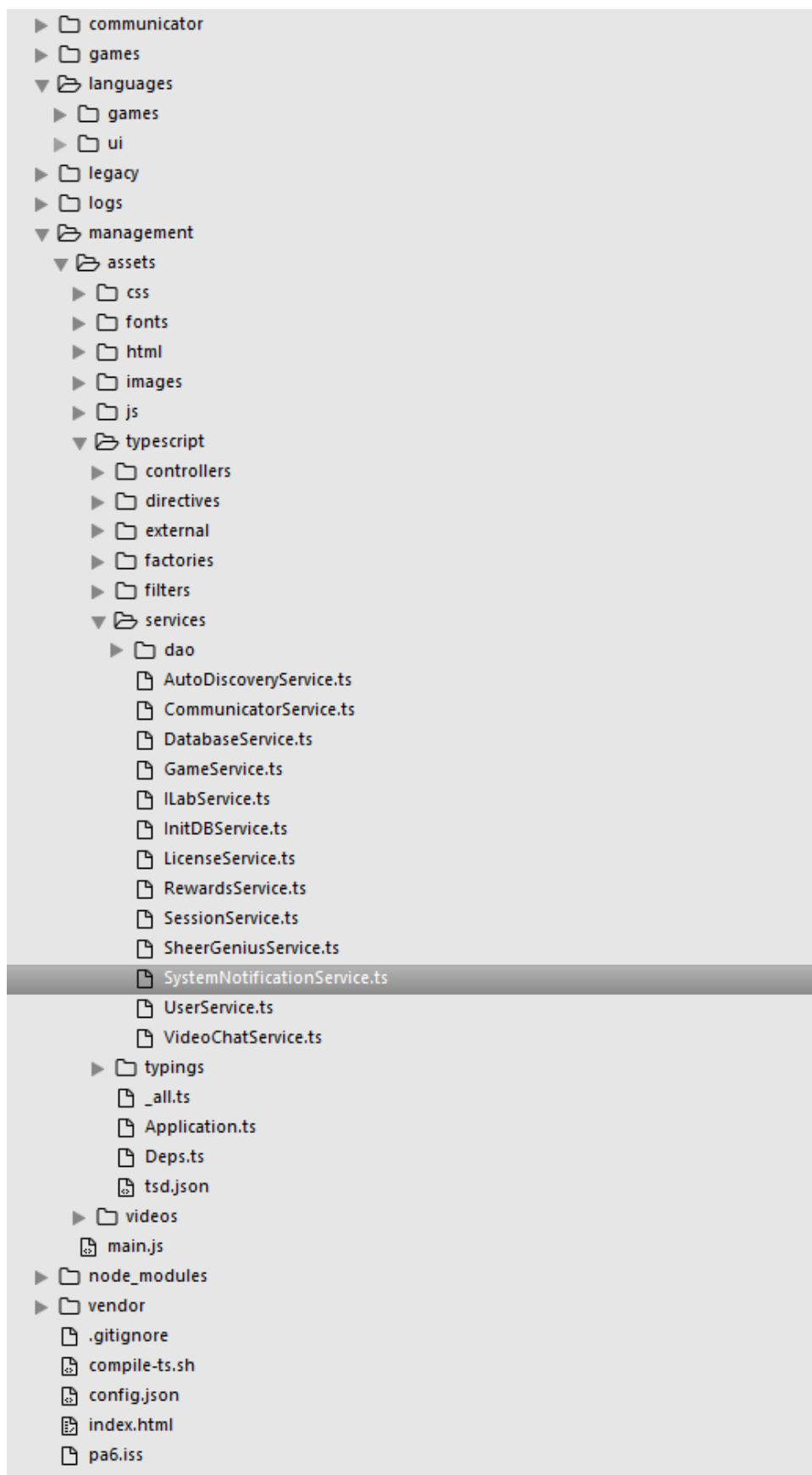


Рисунок 3.9 – Структура проекту

3.6.1 Контролери

Контролери діляться на 4 групи:

- Спільні – використовуються для створення базових компонентів сторінок, наприклад `ApplicationCtrl` виступає у ролі основного контролеру застосунка;
- Студента;
- Тренера;
- Адміністратора.

Основні контролери застосунка, що створені для реалізації базового функціоналу даного проекту:

- `ApplicationCtrl` – виступає у ролі основного контролеру застосунка, він присутній на усіх сторінках;
- `LoginCtrl` – застосовується на сторінці входу у приладову панель, реалізує функцію входу;
- `LogoutCtrl` – застосовується для виходу із приборної панелі користувачів;
- `ProfileCtrl` – застосовується на сторінці профілю користувачів де вони можуть редагувати дані про себе;
- `ReportsCtrl` – контролер включає в себе основні функції для генерації звіту.

3.6.2 Сервіси

Основні сервіси застосунка, що створені для реалізації базового функціоналу даного проекту:

- DAO – низькорівнева обгортка для функцій `LokiJS`;

- DatabaseService – головний сервіс для роботи з LokiJS базою, її колекціями та DAO;
- InitDBService – сервіс, який відповідає за початкову ініціалізацію бази і її даних;
- GameService – сервіс включає в собі усі інтерфейси і функції зв’язані з іграми;
- LicenseService – спостерігає за ліцензією застосунка та функцій, які потрібно обмежати відповідно до типу купленого ключа;
- RewardsService – сервіс включає в собі усі інтерфейси і функції зв’язані з нагородами;
- SessionService – сервіс включає в собі усі інтерфейси і функції зв’язані з сеансами;
- SheerGeniusService – сервіс включає в собі усі інтерфейси і функції штучного інтелекту Sheer Genius;
- SystemNotificationService – сервіс включає в собі усі інтерфейси і функції системних повідомлень для користувачів у різних ситуаціях;
- UserService – сервіс включає в собі усі інтерфейси і функції зв’язані з користувачами.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Підняття проекту

4.1.1 Встановлення необхідних пакетів

Необхідні пакети:

- Система контролю версій Git

Встановлення:

```
$ sudo apt-get install git
```

- Node.js

Встановлення:

```
$ sudo apt-get install nodejs
```

Для створення символічного посилання :

```
$ sudo ln -s `which nodejs` /usr/bin/node
```

- Пакетний менеджер npm

Поставляється разом з пакетом Node.js. Щоб оновити до останньої версії:

```
$ sudo npm i -g npm
```

- Компілятор TypeScript

Встановити TypeScript:

```
$ sudo npm i -g typescript
```

- NW.js версії 0.12.3

Встановлення:

```
$ sudo npm i -g nwjs
```

```
$ nw install 0.12.3
```

4.1.2 Клонування та залежності

Після встановлення усіх необхідних пакетів потрібно скопувати проект з репозиторію:

```
$ git clone <repository>
```

Потім перейти у папку з проектом і встановити усі node-модулі:

```
$ npm install
```

4.1.3 Складання та запуск

Для того щоб зібрати проект треба скомпілювати TypeScript:

```
$ tsc --module commonjs --target ES5 -w --out management/main.js  
management/assets/typescript/_all.ts
```

Після компіляції буде запущено спостерігач, який буде самостійно компілювати TypeScript при зміні файлів.

Для автоматизації компіляції було написано файл compile-ts.sh у корні проекту. Для запуску даного файлу:

```
$ ./compile-ts.sh
```

Для запуску застосування треба перейти у корінь проекту і виконати команду:

```
$ nw .
```

4.2 Відладка

Для відлагодження застосунка був написаний логгер, котрий логує усі виводи в консоль та помилки у папку logs та групує логи по днях.

Код логгера можна знайти у файлі /management/assets/typescript/Deps.ts.

Також можна використовувати WebKit-консоль безпосередньо в застосуванні або інтеграційні та модульні тести.

4.3 Опис функцій

Детальний опис усіх функцій знаходиться безпосередньо в коді.

4.4 Подальша розробка проекту

Так як проект дуже об'ємний та складний подальша розробка проекту без мене неможлива. По всіх питаннях звертатися до мене.

5 КЕРІВНИЦТВО КОРИСТУВАЧА

5.1 Призначення та умови використання програми

Розроблюваний програмний продукт призначений для підвищення уваги, поведінки і когнітивних функцій у дітей хворих на СДУГ.

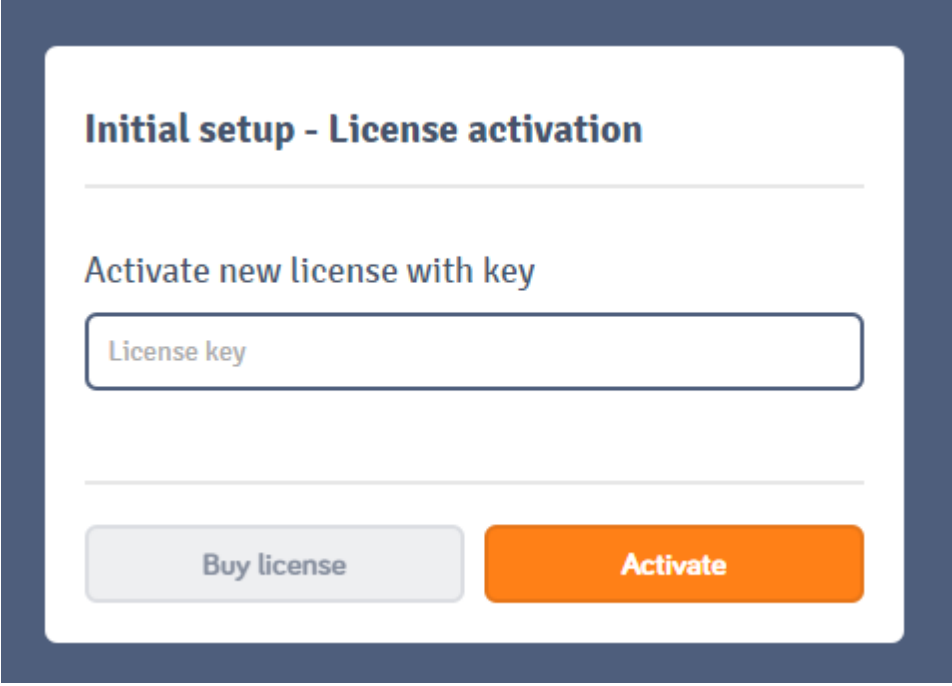
Функції, які мають виконуватися програмним продуктом:

- відстежувати увагу студента при грі;
- можливість проставляти поведінки студента після гри;
- здобування балів за досягнення цілей для витрачання їх на нагороди;
- планувати сеанси для студентів;
- управління студентами та тренерами;
- управління нагородами;
- створення звітів про успіхи студента;
- налаштування застосунка;
- система активації програмного продукту;
- система бекапів/імпорту/експорту;
- підтримка декількох мов.

Для користування розробленим продуктом треба мати 32 або 64 бітну операційну систему Windows, Mac OS X, Linux.

5.2 Виконання програми

Для використання програмного продукту треба його встановити та провести перше налагодження (рис. 5.1 та рис. 5.2).



Initial setup - License activation

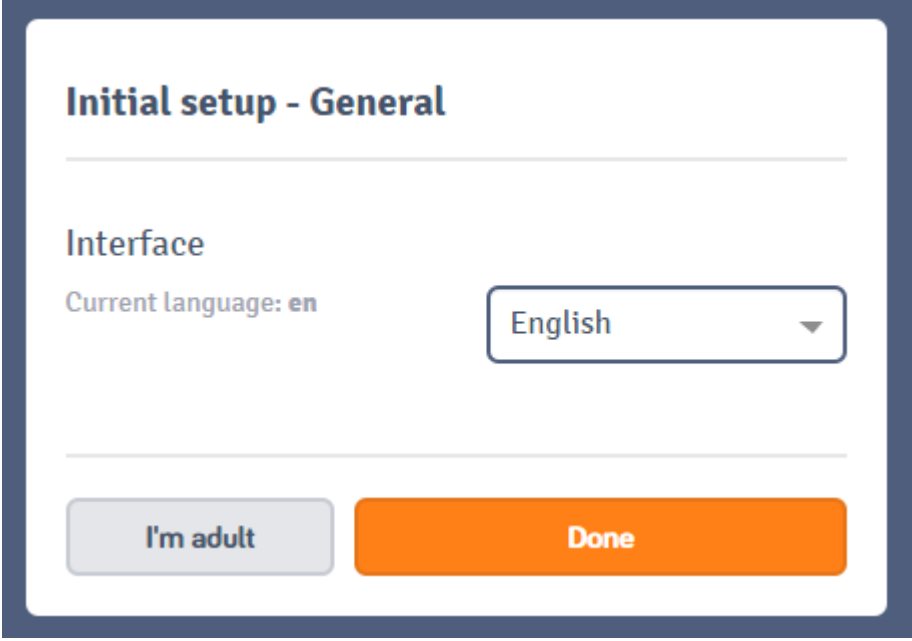
Activate new license with key

License key

Buy license **Activate**

This screenshot shows the 'Initial setup - License activation' screen. It features a title bar at the top, followed by a horizontal separator. Below the separator is the instruction 'Activate new license with key'. A text input field labeled 'License key' is positioned below the instruction. Another horizontal separator follows. At the bottom, there are two buttons: a light gray 'Buy license' button and an orange 'Activate' button.

Рисунок 5.1 – Активація продукту



Initial setup - General

Interface

Current language: en English ▼

I'm adult **Done**

This screenshot shows the 'Initial setup - General' screen. It has a title bar at the top, followed by a horizontal separator. Below the separator is the section header 'Interface'. Under this header, the text 'Current language: en' is displayed next to a dropdown menu currently showing 'English'. A horizontal separator follows. At the bottom, there are two buttons: a light gray 'I'm adult' button and an orange 'Done' button.

Рисунок 5.2 – Головні налаштування

5.3 Використання програми

У програми є 3 типи користувачів: Адміністратор, Тренер та Студент.

5.3.1 Адміністратор

Після першого налагодження адміністратор повинен налаштувати додаткові налаштунки застосування. Для цього йому потрібно перейти на сторінку «Системні налаштування» (рис. 5.3) натиснув на відповідну кнопку з випадаючого меню у верхньому правому кутку програми.

System setup

Interface

Current language: English

English

E-mail settings

Email authentication data

This Email will be used for functions such as Request Progress Review and Email circle of success. Make sure that the settings of this Email are enabled authentication with an unknown devices.

Application uses SMTP protocol for sending Emails

E-mail for sending

E-mail address

Email

Password

Password

E-mail service

E-mail service

Secondary e-mail for Request Progress Review

Email

Devices

Please make sure your BodyWave armband is turned on

Set up BodyWave armband

Save

Рисунок 5.3 – Системні налаштування

На цій сторінці потрібно обов'язково знайти BodyWave-пристрій, щоб студент міг грати в ігри. Також можна налаштувати адрес електронної пошти, щоб можна було використовувати такі функції як: «Запит огляду прогресу» та «Письмо до кола успіху».

Адміністратор може переглядати сеанси усіх студентів або студентів певних тренерів (рис. 5.4).

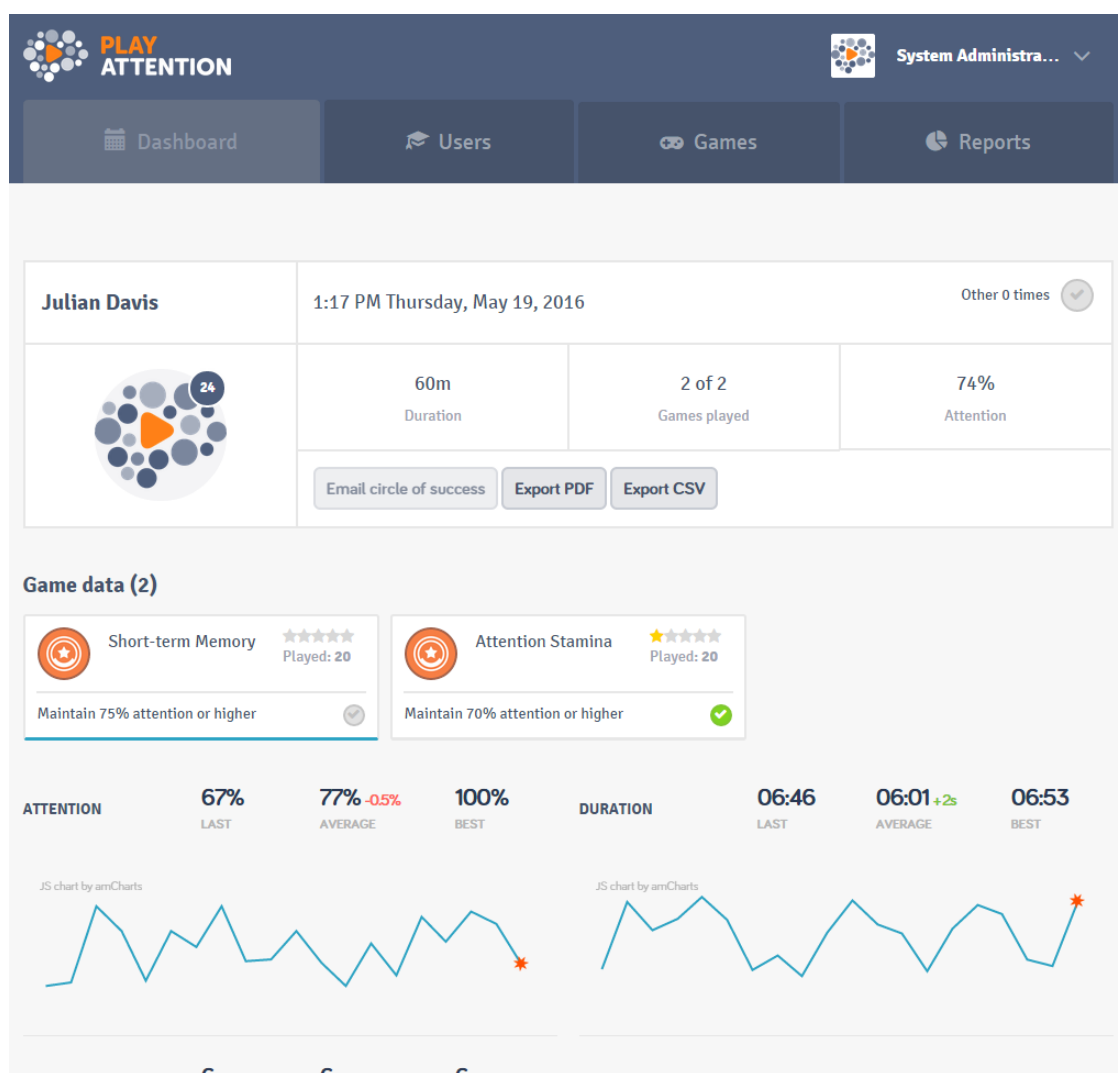


Рисунок 5.4 – Огляд сеансу

Для того, щоб управляти користувачами треба перейти на сторінку «Користувачі» (рис. 5.5) та вибрати користувача серед студентів або тренерів.

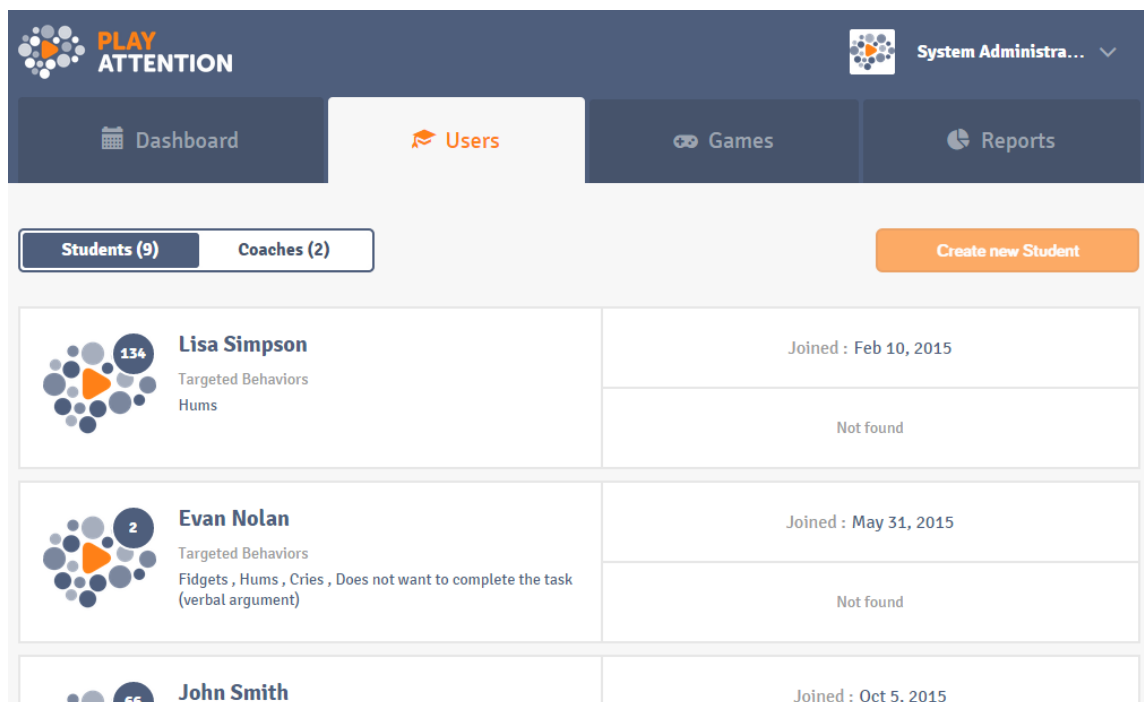


Рисунок 5.5 – Управління користувачами

Також адміністратор може грати, проглядати діючу ліцензію або активувати нову та керувати бекапами/імпортом/експортом (рис. 5.6).

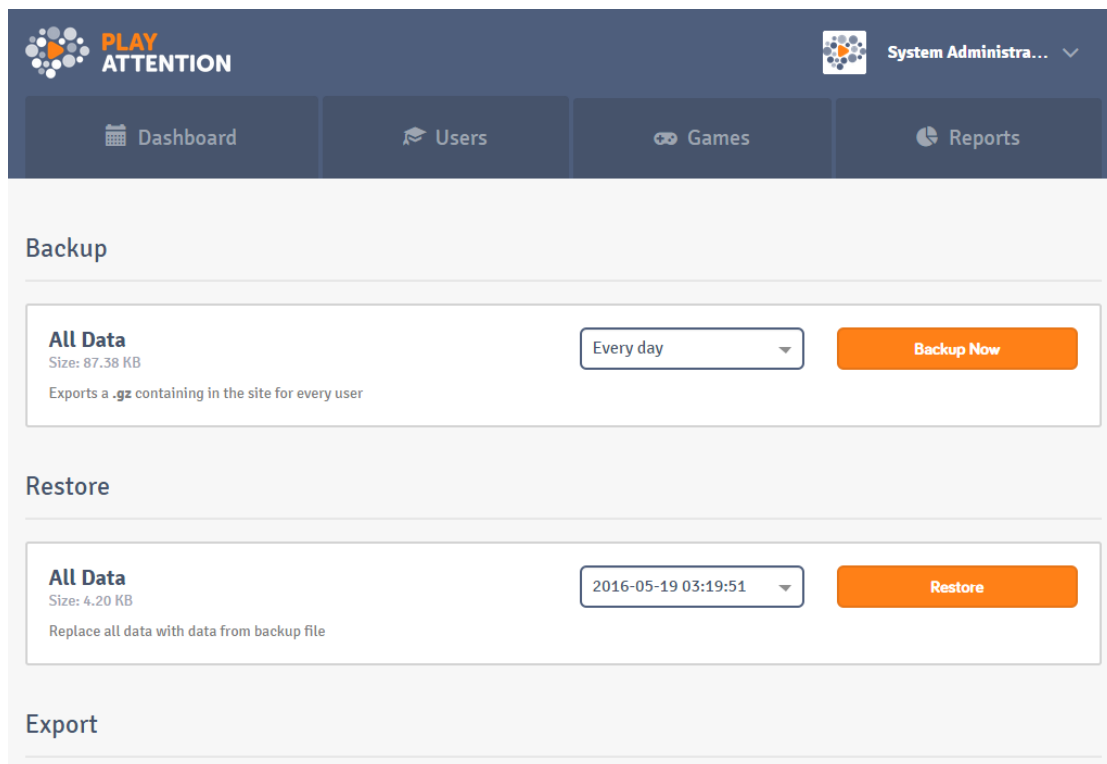


Рисунок 5.6 – Управління бекапами/імпортом/експортом

5.3.2 Тренер

Тренер допомагає направляти студента в процесі навчання і стежити за поведінкою студента в той час, як він грає в ігри. Після кожної гри тренер оцінює певну поведінку, і таким чином вона може бути відслідкована системою (рис. 5.7).

PLAY ATTENTION Fernando Lewis

Behavior Shaping Input Targeted Behaviors All Behaviors

Behavior	Frequency	Target Slider
Off task (e.g. eyes moving off target)	1	Slider at 1
Cannot adjust behavior to expectations of situation	3	Slider at 3
Excitable, impulsive	5	Slider at 5
Does not want to complete the task (verbal argument)	0	Slider at 0
Fails to remember software operations (after 3 sessions)	0	Slider at 0
Other	0	Slider at 0

Show all behaviors

Рисунок 5.7 – Встановлення поведінок тренером після гри студента

Тренер відповідає за створення щотижневого розкладу сеансів (рис. 5.8), а потім проводить кожен з них.

Create New Session

Recurring Session? ☒

Date

05.19.2016

Time

09 :15 AM

Periodicity

Every day

Number of times

3

Duration

00:30

Next session

9:05 am, May, Friday 20

Student

Julian Davis

19

Julian Davis

Joined May 11, 2015

Preview Session as a Student

Reset to Sheer Genius Defaults

Behavioral Objective

Engages in nervous habits (e.g. twists-hair, bite nails, chew objects)

Off task (e.g. eyes moving off target)

Cannot adjust behavior to expectations of situation

Calls out

Fidgets

- Show all behaviors -

Games

Play in order? ☒

Academic Bridge

★★★★★

Time on Task

★★★★★

Discriminatory Processing

★★★★★

Visual Tracking

★★★★★

Select games

Рисунок 5.8 – Створення сеансу

Тренер також створює нагороди (рис. 5.9) та може управляти студентами (рис. 5.10).

Reward name

New Reward

Point value

5

Description

Description for reward

Change picture

☒ Available for all students

Lisa Simpson

Evan Nolan

John Smith

Ada Lovelace

Bjarne Stroustrup

Julian Davis

Today Davis

Othercoach Davis

Unfinished Davis

Save Reward

Рисунок 5.9 – Створення нагороди для студентів

PLAY
ATTENTION

Fernando Lewis

Dashboard

Students

Rewards

Games

Reports

Add New Student

Search students...

My Students

GridList

Lisa Simpson

Joined
February 10, 2015

Targeted Behaviors
Hums

NO SESSIONS

Evan Nolan

Joined
May 31, 2015

Targeted Behaviors
Fidgets , Hums , Cries , Does not want to complete the task (verbal argument)

NO SESSIONS

John Smith

Joined
October 5, 2015

Targeted Behaviors
Fidgets , Hums , Cries

NO SESSIONS

Ada Lovelace

Joined
January 2, 2015

Targeted Behaviors
Fidgets , Hums , Easily frustrated

NO SESSIONS

Рисунок 5.10 – Управління студентами

Додатково тренер може створювати звіти про продуктивність студента і відправити їх до «Кола успіху» студента, якщо це можливо. Тренер так само як і адміністратор має доступ до системних налаштунків.

5.3.3 Студент

Студенти грають сеанси. Сеанс являє собою набір ігор в які студент буде грати за один раз (рис. 5.11).

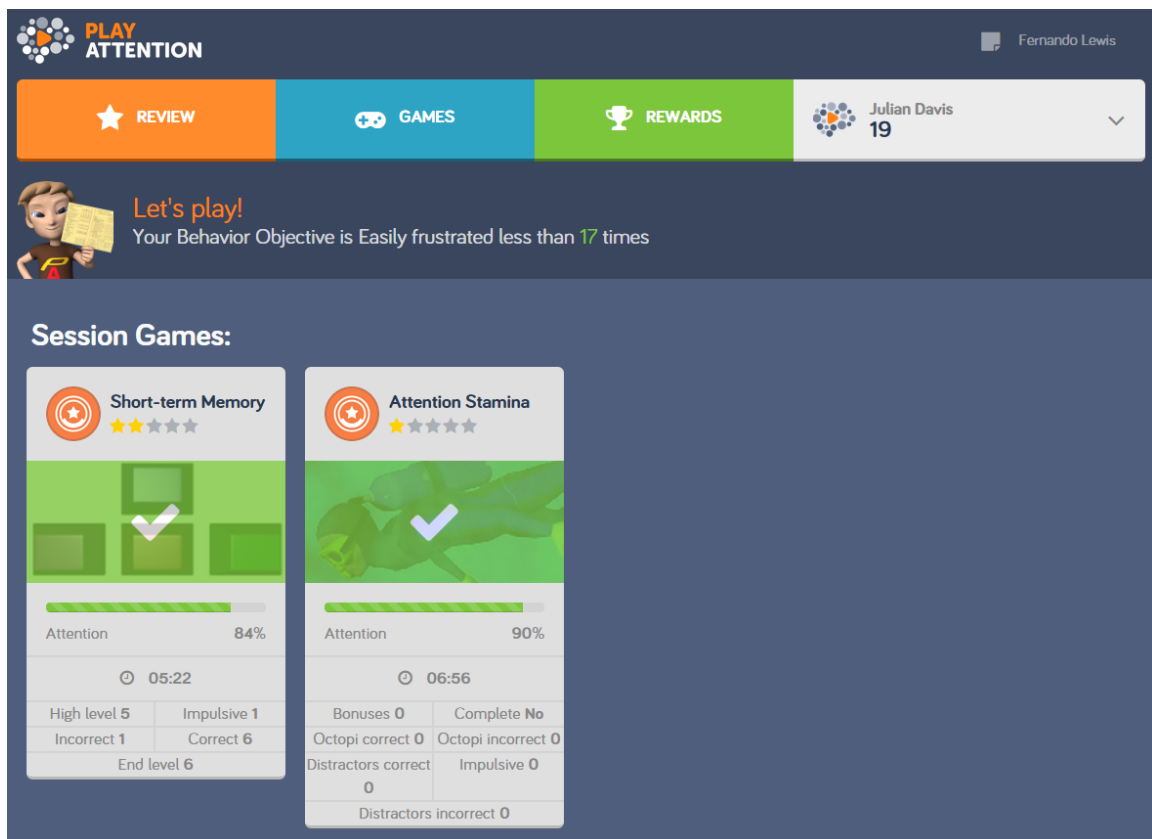


Рисунок 5.11 – Ігри на сеанс

Сеанс починається з огляду попереднього сеансу, потім дозволяє студенту грати в ігри. Завершуючи сеанс, студент має залишити певну замітку щодо пройдених ним ігор (рис. 5.12).

Distraction level

Select a Level ▾

What I've learned

Enter your note here...

What I'm proud of

Enter your note here...

What I need to improve

Short-term Memory: Maintain 75% attention or higher

Attention Stamina: Maintain 75% attention or higher

I'm not done yet Logout

Рисунок 5.12 – Замітка щодо пройдених ігор

У той час, як студент грає в ігри кожного щотижневого сеансу, він може заробити бали в залежності від його продуктивності в іграх. Студент заробляє бали під час досягання цілей, встановлених системою. Бали можуть бути використані для отримання нагород, створених тренером (рис. 5.13).

You are saving points towards these rewards

? Cinema ticket
Enough points to buy!

Choose a reward to save for

Archive My Rewards Buy rewards

1pt ? Pizza party

2pt ? Cinema ticket

Рисунок 5.13 – Придбання нагороди

5.4 Повідомлення оператору

У випадку виникнення небажаних подій, пов'язаних з неправильними діями користувача, некоректно введеними даними та у випадку інших форсмажорних подій програма повідомляє користувача про те, що відбулася відповідна небажана подія, шляхом відображення діалогового вікна або інших засобів з відповідним повідомленням.

ВИСНОВКИ

В результаті виконаної випускної кваліфікаційної роботи бакалавра було створено програмне застосування «Play Attention». Розроблюваний програмний продукт призначений для підвищення уваги, поведінки і когнітивних функцій у дітей хворих на СДУГ.

Функції створеного програмного застосування:

- відстежувати увагу студента при грі;
- можливість проставляти поведінки студента після гри;
- здобування балів за досягнення цілей для витрачання їх на нагороди;
- планувати сеанси для студентів;
- управління студентами та тренерами;
- управління нагородами;
- створення звітів про успіхи студента;
- налаштування застосунка;
- система активації програмного продукту;
- система бекапів/імпорту/експорту;
- підтримка декількох мов.

Для реалізації даного програмного продукту були використані технологія NW.js та фреймворк AngularJS, так як вони значно прискорюють створення аналогічних програмних продуктів та забезпечують стабільність його роботи на багатьох платформах.

Програмне забезпечення створено з використанням технології NW.js, сумісне з x86- та x64-бітними платформами на базі ОС Windows, Mac OS X та Linux. Система має зручний, максимально орієнтований на користувача інтерфейс.

Система була протестована, її відповідність технічному завданню та працездатність підтверджено.

ПЕРЕЛІК ПОСИЛАНЬ

1. СДУГ [Електронний ресурс] – Режим доступу:
<https://uk.wikipedia.org/wiki/%D0%A1%D0%94%D0%A3%D0%93>
2. Play Attention [Електронний ресурс] – Режим доступу:
<http://www.playattention.com/>
3. TypeScript Docs [Електронний ресурс] – Режим доступу:
<https://www.typescriptlang.org/docs/tutorial.html>
4. NW.js Docs [Електронний ресурс] – Режим доступу:
<http://docs.nwjs.io/en/latest/>
5. AngularJS API Docs [Електронний ресурс] – Режим доступу:
<https://docs.angularjs.org/api/>
6. LokiJS Docs [Електронний ресурс] – Режим доступу:
<http://lokijs.org/#/docs>

ДОДАТОК А

ТЕХНІЧНЕ ЗАВДАННЯ

Вступ

Розроблюваний програмний продукт призначений для допомоги людям хворим на СДУГ, щоб поліпшити їх здатність ефективно витратити увагу.

Система будується навколо відносин «тренер-студент». Студент – користувач, який потребує поліпшення здатності ефективно витратити увагу. Використовуючи Play Attention, студент робить це граючи в ігри під час носіння пристрою BodyWave. Пристрій BodyWave забезпечує зворотній зв'язок з застосунком в режимі реального часу, фіксує увагу студента і впливає на гру. Елементи управління на екрані визначають, коли студент може взаємодіяти з застосунком і впливати на процес гри. Тренер допомагає студенту з використанням системи і обговорює з ним (або з представником хворого на СДУГ) його прогрес.

А.1 Підстава для розробки

Підставою для розробки являється завдання на випускню кваліфікаційну роботу бакалавра по Запорізькому Національному Технічному Університету за темою «Розробка програмного забезпечення для тренування дітей з розсіяною увагою» згідно з наказом №202 від 6 квітня 2016 року про затвердження тем, керівників випускних кваліфікаційних робіт бакалаврів та дипломних проектів студентів денної форми навчання освітньо-кваліфікаційного рівня бакалавр напрямів підготовки 6.050103 «Програмна інженерія», 6.050101 «Комп'ютерні науки».

А.2 Призначення для розробки

Розроблюваний програмний продукт призначений для фіксування відхилення уваги дитини у реальному часі під час проходження нею гри. Це повинно допомогти лікарю, вчителю або батькам слідкувати за увагою, поведінкою дитини, і в той же час бути цікавим самій дитині.

А.3 Основні вимоги до розроблюваної програми

А.3.1 Вимоги до функціональних характеристик

Функції, які мають виконуватися програмним продуктом:

- відстежувати увагу студента при грі;
- можливість проставляти поведінки студента після гри;
- здобування балів за досягнення цілей для витрачання їх на нагороди;
- планувати сеанси для студентів;
- управління студентами та тренерами;
- управління нагородами;
- створення звітів про успіхи студента;
- налаштування застосунка;
- система активації програмного продукту;
- система бекапів/імпорту/експорту;
- підтримка декількох мов.

А.3.2 Вимоги до надійності

Програма має забезпечувати коректну обробку помилок у випадку виключних ситуацій.

A.3.3 Умови експлуатації

Для роботи з програмою користувач має володіти базовими знаннями користування комп'ютера.

A.3.4 Вимоги до складу і параметрів технічних засобів

Програма повинна працювати на ПЕОМ із складом та параметрами технічних засобів: 2Gb RAM, мінімум 1.5 Gb вільного місця на жорсткому диску.

A.3.5 Вимоги до інформаційної та програмної сумісності

В склад технічних засобів повинен входити IBM-сумісний персональний комп'ютер. Середовищем функціонування програми є 32 або 64 бітна платформа Windows, Mac OS X та Linux.

A.3.6 Вимоги до програмної документації

Програма має поставлятися з технічним завданням та керівництвом оператора.

A.3.7 Порядок контролю та приймання

Випускна робота допускається до захисту керівником, консультантом та нормоконтролером, затверджується завідуючим кафедрою та захищається у присутності членів Державної екзаменаційної комісії.

ДОДАТОК Б
ТЕКСТ ПРОГРАМИ

/management/assets/Application.ts

```

/// <reference path='_all.ts' />

module app {
    'use strict';

    export interface IRootScope extends ng.IRootScopeService {
        masterAddress:string;
        isIlab:boolean;
        isMaster:boolean;
        $state:any;
        $stateParams:any;
        services:string[];
        isInitd:boolean;
    }

    export enum Platform {
        win32, linux, darwin
    }

    // get app config
    export var appConfig:IConfigManagement = new ConfigManagement();

    export var curPlatform:Platform = Platform[process.platform];
    export var isIlab:boolean = _.isUndefined(appConfig.file['isIlab']) ? false :
Boolean(appConfig.file['isIlab']),
        isMaster:boolean = _.isUndefined(appConfig.file['isMaster']) ? false :
Boolean(appConfig.file['isMaster']);

    export var injector;
    angular.element(document).ready(() => {
        injector = angular.bootstrap(document, ["app.controllers"]);
    });

    angular.module('app.controllers', ['lokijs',
        'ui.router',
        'ui.bootstrap',
        'rzModule',
        'ngSanitize',
        'pascalprecht.translate',
        'ngDraggable',
        'ui.select',
        'ui.sortable',
        'tmh.dynamicLocale',
        'ui.mask'])

    // include factories
    .factory('linker', factories.linker)
    .factory('render', factories.render)

    // include filters

```

```

.filter('chunk', filters.chunk)
.filter('humanizeSeconds', filters.humanizeSeconds)
.filter('humanizeMillis', filters.humanizeMillis)
.filter('humanizeDate', filters.humanizeDate)
.filter('translateFilter', filters.translateFilter)
.filter('tags', filters.tags)
.filter('removeBehaviorPath', filters.removeBehaviorPath)
.filter('count', filters.count)
.filter('except', filters.except)
.filter('removeDotStats', filters.removeDotStats)
.filter('last', filters.last)
.filter('removeWhiteSpaces', filters.removeWhiteSpaces)
.filter('flagged', filters.flagged)
.filter('abs', filters.abs)
.filter('htmlToPlaintext', filters.htmlToPlaintext)
.filter('targetedBehaviors', filters.targetedBehaviors)
.filter('limitObjectTo', filters.limitObjectTo)
.filter('capitalize', filters.capitalize)
.filter('escape', filters.escape)

// include directives
.directive('togglepass', directives.togglepass)
.directive('fullcalendar', directives.fullcalendar)

// include services
.service('databaseService', ['Loki', '$q', (l, $q) => new DatabaseService(l, $q)])
.service('initDBService', ['databaseService', 'gameService', 'sheerGeniusService',
'$rootScope', 'sessionService',
(db, gs, sgs, $r, ss) => new InitDBService(db, gs, sgs, $r, ss)])
.service('userService', ['databaseService', '$rootScope', (db, $r) => new
UserService(db, $r)])
.service('gameService', ['databaseService', 'userService', (db, us) => new
GameService(db, us)])
.service('sessionService', ['databaseService', '$filter', 'gameService', '$state',
'$rootScope', 'videoChatService',
(db, $f, gs, $s, $r, vcs) => new SessionService(db, $f, gs, $s, $r, vcs)])
.service('rewardsService', ['databaseService', 'userService', '$filter',
(db, us, $f) => new RewardsService(db, us, $f)])
.service('sheerGeniusService', ['gameService', 'sessionService', 'userService',
(gs, ss, us) => new SheerGeniusService(gs, ss, us)])
.service('autoDiscoveryService', ['$rootScope', 'databaseService', '$timeout',
'$filter', 'videoChatService',
($r, db, $t, $f, vcs) => new AutoDiscoveryService($r, db, $t, $f, vcs)])
.service('iLabService', ['autoDiscoveryService', 'userService', '$rootScope',
'gameService', '$http',
'$interval', 'sessionService', '$filter',
(ads, us, $r, gs, $http, $interval, ss, $f) => new ILabService(ads, us, $r, gs,
$http, $interval, ss, $f)])
.service('videoChatService', ['$rootScope', '$q', ($r, $q) => new
VideoChatService($r, $q)])
.service('systemNotificationService', ['databaseService', '$rootScope', (db, $r) =>
new SystemNotificationService(db, $r)])

```

```

        .service('communicatorService', ['$filter', '$rootScope', ($f, $r) => new
CommunicatorService($f, $r)])
        .service('licenseService', ['$rootScope', '$modal', 'userService', ($r, $m, us) =>
new LicenseService($r, $m, us)])

// include controllers
.controller('ApplicationCtrl', ApplicationCtrl)
.controller('ReviewCtrl', ReviewCtrl)
.controller('GamesCtrl', GamesCtrl)
.controller('RewardsCtrl', RewardsCtrl)
.controller('RewardsModalCtrl', RewardsModalCtrl)
.controller('ProfileCtrl', ProfileCtrl)
.controller('LogoutCtrl', LogoutCtrl)
.controller('GameResultsCtrl', GameResultsCtrl)
.controller('BehaviorInputCtrl', BehaviorInputCtrl)
.controller('WaitBehaviorCtrl', WaitBehaviorCtrl)
.controller('CurrentResultCtrl', CurrentResultCtrl)
.controller('CoachDashboardCtrl', CoachDashboardCtrl)
.controller('CoachStudentEditCtrl', CoachStudentEditCtrl)
.controller('CoachStudentsCtrl', CoachStudentsCtrl)
.controller('LoginCtrl', LoginCtrl)
.controller('GameResultsHeaderModalCtrl', GameResultsHeaderModalCtrl)
.controller('ComponentsCtrl', ComponentsCtrl)
.controller('SelectGameModalCtrl', SelectGameModalCtrl)
.controller('CoachStudentInfoCtrl', CoachStudentInfoCtrl)
.controller('SelectBehaviorModalCtrl', SelectBehaviorModalCtrl)
.controller('CreateSessionModalCtrl', CreateSessionModalCtrl)
.controller('GiveRewardModalCtrl', GiveRewardModalCtrl)
.controller('ConfirmModalCtrl', ConfirmModalCtrl)
.controller('CoachRewardsCtrl', CoachRewardsCtrl)
.controller('CoachEditRewardCtrl', CoachEditRewardCtrl)
.controller('CoachSessionReviewCtrl', CoachSessionReviewCtrl)
.controller('CoachGamesCtrl', CoachGamesCtrl)
.controller('CoachGameInfoCtrl', CoachGameInfoCtrl)
.controller('ReportsCtrl', ReportsCtrl)
.controller('AdminUsersCtrl', AdminUsersCtrl)
.controller('AdminCoachInfoCtrl', AdminCoachInfoCtrl)
.controller('AdminImportExportCtrl', AdminImportExportCtrl)
.controller('AdminTestingToolsCtrl', AdminTestingToolsCtrl)
.controller('SystemSetupCtrl', SystemSetupCtrl)
.controller('CoachILabCtrl', CoachILabCtrl)
.controller('CoachILabStatsCtrl', CoachILabStatsCtrl)
.controller('CoachILabBehaviorTrackingCtrl', CoachILabBehaviorTrackingCtrl)
.controller('ChatCtrl', ChatCtrl)
.controller('RequestProgressReviewCtrl', RequestProgressReviewCtrl)
.controller('EmailCircleOfSuccessCtrl', EmailCircleOfSuccessCtrl)
.controller('LoginChoiceModalCtrl', LoginChoiceModalCtrl)
.controller('ConfigAppCtrl', ConfigAppCtrl)
.controller('SetGameLevelModalCtrl', SetGameLevelModalCtrl)
.controller('CongratulationsCtrl', CongratulationsCtrl)
.controller('StudentHeaderCtrl', StudentHeaderCtrl)
.controller('LicenseModalCtrl', LicenseModalCtrl)

```

```

.controller('AdminLicenseCtrl', AdminLicenseCtrl)

.config(['$stateProvider', '$urlRouterProvider', '$translateProvider',
'$translatePartialLoaderProvider',
    'tmhDynamicLocaleProvider',
    function ($stateProvider, $urlRouterProvider, $translateProvider,
$translatePartialLoaderProvider,
        tmhDynamicLocaleProvider) {
    // Localization parts
    // TODO: do async
    const translationParts = ['ui', 'games'];

    try {
        _.forEach(translationParts, part => {
            $translatePartialLoaderProvider.addPart(part);

            var partDir = fs.readdirSync('languages/' + part);

            _.forEach(partDir, file => {
                var filePath = 'languages/' + part + '/' + file;

                if(node_path.extname(filePath) === '.csv'){
                    var CSVData = fs.readFileSync(filePath, 'utf-8'),
                        splitedData = CSVData.split('\n'),
                        translation = {};

                    _.forEach(splitedData, row => {
                        var splitedRow = row.split('\t');

                        // key: value
                        translation[splitedRow[0]] = splitedRow[1];
                    });

                    fs.writeFileSync(filePath.replace('.csv', '.json'),
JSON.stringify(translation));
                }
            });
        });
    } catch (e) {
        console.error(e);
    }

    // JSON to CSV
    /*
    _.forEach(translationParts, part => {
        var partDir = fs.readdirSync('languages/' + part);

        _.forEach(partDir, file => {
            var filePath = 'languages/' + part + '/' + file;

            if(node_path.extname(filePath) === '.json'){

```

```

        var jsonData = JSON.parse(fs.readFileSync('languages/' + part +
        '/' + file)),
        joinData = _.map(jsonData, (val, key) => key + '\t' +
val).join('\n');

        fs.writeFileSync('languages/' + part + '/' +
(file).replace('.json', '.csv'), joinData);
    }
    });
    });
    */

    // Localization default settings
    $translateProvider.useLoader('$translatePartialLoader', {
        urlTemplate: 'languages/{part}/locale-{lang}.json'
    })
    .preferredLanguage(appConfig ? appConfig.file['app-language'] : 'en')
    // About sanitize:
    // http://stackoverflow.com/questions/31002499/angular-translate-
sanitisation-fails-with-utf-characters
    .useSanitizeValueStrategy('sanitizeParameters');
    // Set location for Angular locale files
    tmhDynamicLocaleProvider
    .localeLocationPattern("management/assets/js/angular-locales/angular-
locale_{{locale}}.js");

    // fix for https://github.com/angular-ui/ui-router/issues/600
    $urlRouterProvider.otherwise( ($injector, $location) => {
        var $state = $injector.get("$state");

        // config app for first launch
        if ( _.isUndefined(appConfig.file['first-launch']) ||
appConfig.file['first-launch'] ) {
            $state.go('config_app');
        } else {
            $state.go('login');
        }
    });

    $stateProvider
    .state('login', {
        url: "/login",
        views: {
            "app@": {
                templateUrl: "management/assets/html/login.html",
                controller: LoginCtrl
            }
        },
        data: {
            hideNav: true
        },
        resolve: {

```

```

        dbService: (databaseService) => {
            return databaseService.deferred.promise;
        }
    })
    .state('config_app', {
        url: "/config_app",
        views: {
            "app@": {
                templateUrl: "management/assets/html/config-app.html",
                controller: ConfigAppCtrl
            }
        }
    })
    .state('app', {
        url: "/app",
        views: {
            "app@": {
                templateUrl: "management/assets/html/student/app.html",
                controller: ApplicationCtrl
            },
            "chat@app": {
                templateUrl: "management/assets/html/chat.html",
                controller: ChatCtrl
            },
            "video-chat@app": {
                templateUrl: "management/assets/html/video-chat.html",
                controller: VideoChatCtrl
            },
            "header@app": {
                templateUrl: "management/assets/html/student/header.html",
                controller: StudentHeaderCtrl
            }
        }
    })
    .state('app.review', {
        url: "/review",
        views: {
            "content@app": {
                templateUrl:
"management/assets/html/student/review/review.html",
                controller: ReviewCtrl
            }
        }
    })
    .state('app.games', {
        url: "/games",
        views: {
            "content@app": {
                templateUrl:
"management/assets/html/student/games/games.html",
                controller: GamesCtrl
            }
        }
    })

```

```

        }
    }
})
.state('app.games.game', {
    url: '/:game',
    views: {
        "content@app": {
            templateUrl: "management/assets/html/student/games/game-
results.html",

            controller: GameResultsCtrl
        }
    }
})
.state('app.games.game.behavior', {
    url: '/behavior-input',
    views: {
        "content@app": {
            templateUrl:
"management/assets/html/student/games/behavior-input.html",
            controller: BehaviorInputCtrl
        },
        "header@app": {}
    },
    data: {
        hideNav: true
    }
})
.state('app.games.game.waitBehavior', {
    url: '/wait-behavior',
    views: {
        "content@app": {
            templateUrl: "management/assets/html/student/games/wait-
behavior.html",

            controller: WaitBehaviorCtrl
        },
        "header@app": {}
    },
    data: {
        hideNav: true
    }
})
.state('app.games.game.currentResult', {
    url: '/current-result',
    views: {
        "content@app": {
            templateUrl: "management/assets/html/student/games/current-
results.html",

            controller: CurrentResultCtrl
        }
    }
})
.state('app.games.game.congratulations', {

```

```

        url: '/congratulations',
        views: {
            "content@app": {
                templateUrl:
"management/assets/html/student/games/congratulations.html",
                controller: CongratulationsCtrl
            },
            "header@app": {}
        },
        data: {
            indentMenu: true
        }
    })
    .state('app.rewards', {
        url: "/rewards",
        views: {
            "content@app": {
                templateUrl:
"management/assets/html/student/rewards/rewards.html",
                controller: RewardsCtrl
            }
        }
    })
    .state('app.profile', {
        url: "/profile",
        views: {
            "content@app": {
                templateUrl:
"management/assets/html/student/profile/profile.html",
                controller: ProfileCtrl
            },
            "header@app": {
                templateUrl:
"management/assets/html/student/profile/profile-header.html"
            }
        }
    })
    .state('app.logout', {
        url: "/logout",
        views: {
            "content@app": {
                templateUrl:
"management/assets/html/student/logout/logout.html",
                controller: LogoutCtrl
            }
        }
    })

    .state('coach', {
        url: "/coach",
        views: {
            "app@": {

```

```

        templateUrl: "management/assets/html/coach/app.html",
        controller: ApplicationCtrl
    },
    "chat@coach": {
        templateUrl: "management/assets/html/chat.html",
        controller: ChatCtrl
    },
    "video-chat@coach": {
        templateUrl: "management/assets/html/video-chat.html",
        controller: VideoChatCtrl
    }
}
}))

.state('coach.dashboard', {
    url: "/dashboard",
    views: {
        "content@coach": {
            templateUrl:
"management/assets/html/coach/dashboard/dashboard.html",
            controller: CoachDashboardCtrl
        }
    }
})

.state('coach.settings', {
    url: "/settings",
    views: {
        "content@coach": {
            templateUrl: "management/assets/html/coach/settings.html",
            controller: ProfileCtrl
        }
    }
})

.state('coach.dashboard.session', {
    url: '/:session',
    views: {
        "content@coach": {
            templateUrl: "management/assets/html/coach/dashboard/edit-
session.html",
            controller: CoachSessionEditCtrl
        }
    },
    params: {
        sessionObject: null
    }
})

.state('coach.students', {
    url: "/students",
    views: {
        "content@coach": {

```

```

        templateUrl:
"management/assets/html/coach/students/students.html",
        controller: CoachStudentsCtrl
    }
}
}))
.state('coach.students.show', {
    url: '/show/{student:int}',
    views: {
        "content@coach": {
            templateUrl:
"management/assets/html/coach/students/student-info.html",
            controller: CoachStudentInfoCtrl
        }
    }
})
.state('coach.students.edit', {
    url: '/edit/:student',
    views: {
        "content@coach": {
            templateUrl: "management/assets/html/coach/students/edit-
student.html",
            controller: CoachStudentEditCtrl
        }
    }
})
.state('coach.rewards', {
    url: '/rewards',
    views: {
        "content@coach": {
            templateUrl:
"management/assets/html/coach/rewards/rewards.html",
            controller: CoachRewardsCtrl
        }
    }
})
.state('coach.rewards.reward', {
    url: '/:reward',
    views: {
        'content@coach': {
            templateUrl:
'management/assets/html/coach/rewards/reward.html',
            controller: CoachEditRewardCtrl
        }
    }
})
.state('coach.games', {
    url: '/games',
    views: {
        "content@coach": {
            templateUrl:
"management/assets/html/coach/games/games.html",

```

```

        controller: CoachGamesCtrl
    }
}
}))
.state('coach.games.show', {
    url: '/show/:game',
    views: {
        "content@coach": {
            templateUrl: "management/assets/html/coach/games/game-
info.html",

            controller: CoachGameInfoCtrl
        }
    }
}))
.state('coach.reports', {
    url: '/reports',
    views: {
        "content@coach": {
            templateUrl: "management/assets/html/reports.html",
            controller: ReportsCtrl
        }
    },
    params: {
        id: null
    }
}))

.state('coach.session_review', {
    url: '/session_review',
    views: {
        'content@coach': {
            templateUrl: "management/assets/html/coach/session-
review.html",

            controller: CoachSessionReviewCtrl
        }
    },
    params: {
        session: null,
        student: null
    }
}))

.state('coach.system_setup', {
    url: "/system_setup",
    views: {
        "content@coach": {
            templateUrl: "management/assets/html/system-setup.html",
            controller: SystemSetupCtrl
        }
    }
}))

```

```

.state('coach.communicator', {
  url: "/communicator",
  views: {
    "content@coach": {
      templateUrl: "management/assets/html/communicator.html",
      controller: CommunicatorCtrl
    }
  }
})

.state('coach.ilab', {
  url: '/ilab?list',
  views: {
    "content@coach": {
      templateUrl: "management/assets/html/coach/ilab/ilab.html",
      controller: CoachILabCtrl
    },
  }
})

.state('coach.ilab.stats', {
  url: '/stats/{id:int}',
  views: {
    "content@coach": {
      templateUrl:
"management/assets/html/coach/ilab/stats.html",
      controller: CoachILabStatsCtrl
    },
  }
})

.state('coach.ilab.behavior_tracking', {
  url: '/behavior_tracking/{id:int}',
  views: {
    "content@coach": {
      templateUrl: "management/assets/html/coach/ilab/behavior-
tracking.html",
      controller: CoachILabBehaviorTrackingCtrl
    },
  }
})

.state('components', {
  url: '/components',
  views: {
    'app@': {
      templateUrl: 'management/assets/html/components.html',
      controller: ComponentsCtrl
    }
  }
})

```

```

        .state('admin', {
            url: "/admin",
            views: {
                "app@": {
                    templateUrl: "management/assets/html/admin/app.html",
                    controller: ApplicationCtrl
                }
            }
        })
        .state('admin.dashboard', {
            url: "/dashboard",
            views: {
                "content@admin": {
                    templateUrl:
"management/assets/html/coach/dashboard/dashboard.html",
                    controller: CoachDashboardCtrl
                }
            }
        })
        .state('admin.dashboard.session', {
            url: '/:session',
            views: {
                "content@admin": {
                    templateUrl: "management/assets/html/coach/dashboard/edit-
session.html",

                    controller: CoachSessionEditCtrl
                }
            },
            params: {
                plannedDatetime: null
            }
        })

        .state('admin.users', {
            url: "/users",
            views: {
                "content@admin": {
                    templateUrl:
"management/assets/html/admin/users/users.html",
                    controller: AdminUsersCtrl
                }
            }
        })
        .state('admin.show_student', {
            url: '/students/show/{student:int}',
            views: {
                "content@admin": {
                    templateUrl:
"management/assets/html/coach/students/student-info.html",
                    controller: CoachStudentInfoCtrl
                }
            }
        })

```

```

    }
  })
  .state('admin.edit_student', {
    url: '/students/edit/:student',
    views: {
      "content@admin": {
        templateUrl: "management/assets/html/coach/students/edit-
student.html",

        controller: CoachStudentEditCtrl
      }
    }
  })
  .state('admin.show_coach', {
    url: '/coaches/show/{id:int}',
    views: {
      "content@admin": {
        templateUrl: "management/assets/html/admin/users/coach-
info.html",

        controller: AdminCoachInfoCtrl
      }
    }
  })
  .state('admin.edit_coach', {
    url: '/coaches/edit/:id',
    views: {
      "content@admin": {
        templateUrl: "management/assets/html/admin/users/edit-
coach.html",

        controller: AdminCoachEditCtrl
      }
    }
  })

  .state('admin.games', {
    url: '/games',
    views: {
      "content@admin": {
        templateUrl:
"management/assets/html/coach/games/games.html",
        controller: CoachGamesCtrl
      }
    }
  })
  .state('admin.games.show', {
    url: '/show/:game',
    views: {
      "content@admin": {
        templateUrl: "management/assets/html/coach/games/game-
info.html",

        controller: CoachGameInfoCtrl
      }
    }
  })

```

```

    })
    .state('admin.reports', {
      url: '/reports',
      views: {
        'content@admin': {
          templateUrl: "management/assets/html/reports.html",
          controller: ReportsCtrl
        }
      },
      params: {
        id: null
      }
    })
    .state('admin.session_review', {
      url: '/session_review',
      views: {
        'content@admin': {
          templateUrl: "management/assets/html/coach/session-
review.html",

          controller: CoachSessionReviewCtrl
        }
      },
      params: {
        session: null,
        student: null
      }
    })
    .state('admin.import_export', {
      url: '/import_export',
      views: {
        'content@admin': {
          templateUrl: "management/assets/html/admin/import-
export.html",

          controller: AdminImportExportCtrl
        }
      }
    })
    .state('admin.testing_tools', {
      url: '/testing_tools',
      views: {
        'content@admin': {
          templateUrl: "management/assets/html/admin/testing-
tools.html",

          controller: AdminTestingToolsCtrl
        }
      }
    })
    .state('admin.system_setup', {
      url: "/system_setup",
      views: {
        "content@admin": {
          templateUrl: "management/assets/html/system-setup.html",

```

```

        controller: SystemSetupCtrl
    }
}
}))
.state('admin.communicator', {
    url: "/communicator",
    views: {
        "content@admin": {
            templateUrl: "management/assets/html/communicator.html",
            controller: CommunicatorCtrl
        }
    }
})
.state('admin.license', {
    url: "/license",
    views: {
        "content@admin": {
            templateUrl: "management/assets/html/admin/license.html",
            controller: AdminLicenseCtrl
        }
    }
})
}))

.run(['$rootScope', '$state', '$stateParams', '$urlRouter', 'initDBService',
'gameService',
'iLabService', 'databaseService', 'communicatorService', 'licenseService',
($rootScope:IRootScope,
$state,
$stateParams,
$urlRouter,
initDB: IInitDBService,
gameService: IGameService,
iLabService: IiLabService,
databaseService: IDatabaseService,
communicatorService: ICommunicatorService,
licenseService: ILicenseService) => {
    $rootScope.masterAddress = null;
    $rootScope.isIlab = isIlab;
    $rootScope.isMaster = isMaster;
    $rootScope.$state = $state;
    $rootScope.$stateParams = $stateParams;
    $rootScope.services = [];

    $rootScope.$on('$stateChangeStart',
        (event, toState, toParams, fromState, fromParams) => {
            // wait master address for iLab
            if ( isIlab && !isMaster && _.isNull($rootScope.masterAddress) ) {
                event.preventDefault();
                $rootScope.$on('masterAddressChanged', () => {
                    $state.go(toState, toParams);
                });
            }
        });
    });

```

```

    });
  }

  // load game service
  if ( !(isIlab && !isMaster) && toState.name !== 'login' &&
!gameService.loaded ){

    event.preventDefault();
    databaseService.loadPromise.then(() => {
      gameService.reInit().then(() => {
        $state.go(toState, toParams);
      });
    });
  }
});

const studentThemes = {
  child: 'childTheme',
  teen: 'teenTheme',
  adult: 'adultTheme'
};

$rootScope.$on('$stateChangeSuccess', () => {
  if (UserService.getCurrentUser() && UserService.getCurrentUser().role
=== Role.student){

    var body = $('body');
    switch( UserService.getCurrentUser().age){
      case Age.child:
        body.addClass(studentThemes.child);
        break;
      case Age.teen:
        body.addClass(studentThemes.teen);
        break;
      case Age.adult:
        body.addClass(studentThemes.adult);
        break;
      default:
        body.addClass(studentThemes.child);
        break;
    }
  } else {
    _.forEach(studentThemes, (theme, key) => {
      $('body').removeClass(theme);
    });
  }
});

window.onbeforeunload = () => {
  communicatorService.stop();
};

// Remove old session data when user close the app before logout
win.on('close', function(){

```

```
        if(localStorage){
            localStorage.removeItem('user');
            localStorage.removeItem('session');
        }
        communicatorService.stop();
        this.close(true);
    });

    communicatorService.start();
    $rootScope.isInitiated = true;
    $rootScope.$broadcast('app:initiated');
    });
}
```

ДОДАТОК В

СЛАЙДИ ПРЕЗЕНТАЦІЇ

Національний університет «Запорізька політехніка»

кафедра СА та ОМ

Випускна кваліфікаційна робота бакалавра

**Розробка програмного забезпечення для
тренування дітей з розсіяною увагою**

Виконав
ст. гр. КНТ-818сп

В. Р. Ворона

Керівник

А. Є. Рябенко

Об'єктом дослідження випускної кваліфікаційної роботи є процес організації роботи застосунка.

Предмет дослідження – фіксувати відхилення уваги дитини у реальному часі під час проходження нею гри.

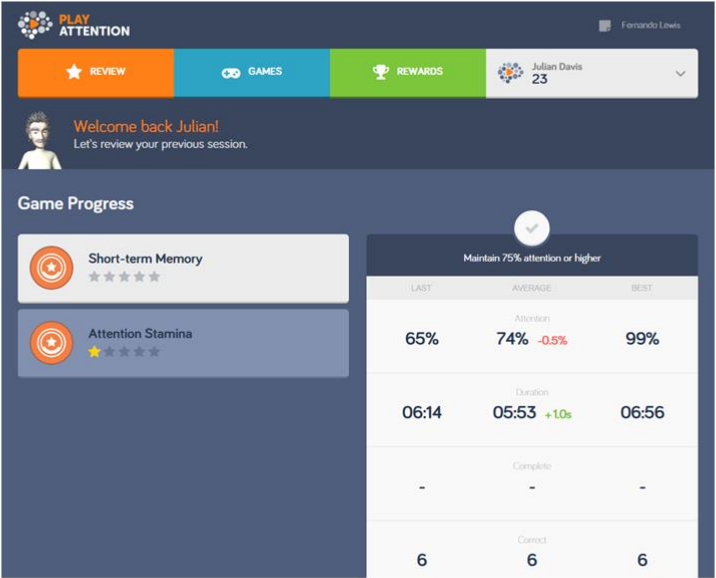
Мета роботи – створення програмного забезпечення для підвищення уваги, поведінки і когнітивних функцій у дітей та дорослих з синдромом дефіциту уваги і гіперактивності.



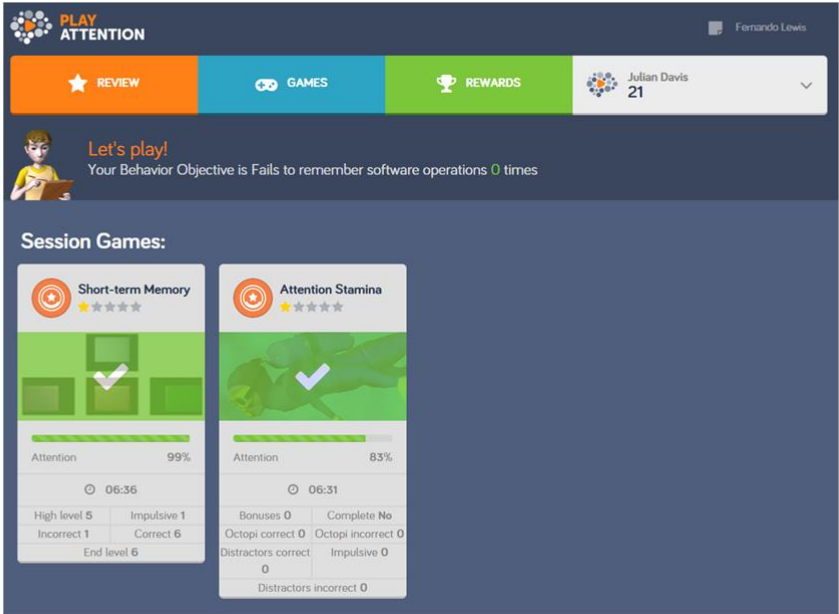
Вхід до приладової панелі

A login form for 'PLAY ATTENTION' on a dark blue background. At the top is the logo, which consists of a cluster of grey dots with an orange play button icon in the center, followed by the text 'PLAY ATTENTION' in orange and white. Below the logo are two input fields: 'Username' and 'Password'. Underneath these fields is a large orange 'Login' button. In the bottom right corner, there is a small link that says 'Like admin?'.

Приладова панель студента



Ігри на сеанс студента



Гра «Attention Stamin»



Сторінка виходу із сеансу

Distraction level

Select a Level ▾

What I've learned

Enter your note here...

What I'm proud of

Enter your note here...

What I need to improve
Short-term Memory: Maintain 75% attention or higher
Attention Stamina: Maintain 75% attention or higher

I'm not done yet

Logout

Розклад сеансів

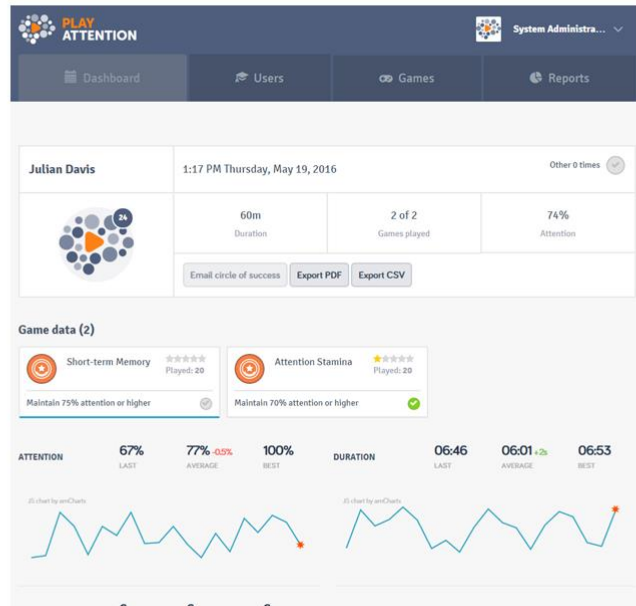
The screenshot displays the 'PLAY ATTENTION' dashboard for user 'Fernando Lewis'. The main section is a calendar for May 2016, showing sessions for 'Julian' at 8:39 AM. A pop-up window for '8:39 AM, Saturday, 7 May' provides details for 'Julian Davis', including a 'Fails to remember software operations 0 times' status and performance metrics for 'Short-term Memory' (Bronze) and 'Attention Stamina' (Bronze). The interface includes a top navigation bar with 'Dashboard', 'Students', 'Rewards', 'Games', and 'Reports'.

Створення сеансу

The 'Create New Session' form includes the following fields and sections:

- Date:** 05.19.2016
- Time:** 09:15 AM
- Periodicity:** Every day
- Number of times:** 3
- Recurring Session?:** ☒
- Duration:** 00:30
- Next session:** 9:05 am, May, Friday 20
- Student:** Julian Davis (Joined May 11, 2015)
- Behavioral Objective:**
 - Engages in nervous habits (e.g. twists-hair, bite nails, chew objects)
 - Off task (e.g. eyes moving off target)
 - Cannot adjust behavior to expectations of situation
 - Calls out
 - Fidgets
- Games:**
 - Academic Bridge (★★★★★)
 - Time on Task (★★★★★)
 - Discriminatory Processing (★★★★★)
 - Visual Tracking (★★★★★)

Звіт сеансу



Системні налаштунки

System setup

Interface
Current language: English English

E-mail settings

Email authentication data
This Email will be used for functions such as Request Progress Review and Email circle of success. Make sure that the settings of this Email are enabled authentication with an unknown devices.
Application uses SMTP protocol for sending Emails

E-mail for sending

E-mail address

Password Show

E-mail service

Secondary e-mail for Request Progress Review

Devices

Please make sure your BodyWave armband is turned on

[Set up BodyWave armband](#)

Дякую за увагу !!!

