UNIWERSYTET ŁÓDZKI
WYDZIAŁ EKONOMICZNO-SOCJOLOGICZNY
STUDIA STACJONARNE I STOPNIA
KIERUNEK: INFORMATYKA EKONOMICZNA

WOJCIECH BIBEL
NR ALBUMU: 390259

MOOD TRACKER Z ELEMENTAMI WCZESNEGO OSTRZEGANIA PRZED ZABURZENIAMI ODŻYWIANIA.

Praca licencjacka napisana
Pod kierunkiem naukowym
dr Marty R. Jabłońskiej
Katedra Informatyki Ekonomicznej i Medycznej

Łódź, 2023

SPIS TREŚCI

WSTĘP

Temat zaburzeń odżywiania stanowi istotny aspekt zdrowia publicznego, ponieważ dotyczy on coraz większej liczby osób, zarówno w krajach rozwiniętych, jak i rozwijających się. Odnotowuje się stały wzrost liczby osób cierpiących na zdefiniowane bądź niezdefiniowane różnego rodzaju zaburzenia odżywiania, a liczba ta stale rośnie, do czego przyczynił się okres pandemiczny[1]. Zaburzenia odżywiania, takie jak anoreksja, bulimia lub jedzenie obsesyjno-kompulsywne, są często bardzo trudne do zdiagnozowania i leczenia, a także mogą prowadzić do poważnych problemów zdrowotnych, a nawet śmierci[2]. Dążenia do poprawy wyglądu zewnętrznego są głęboko zakorzenione w społeczeństwie, a zwłaszcza u kobiet[3].

W ostatnich latach aplikacje mobilne przeżywają znaczący rozwój. Media społecznościowe takie jak np. Instagram, Tiktok, Facebook czy YouTube zdominowały Internet i są głównym źródłem informacji i wzorców zwłaszcza dla młodych ludzi[4]. Bardzo często prezentowane tam sylwetki wysportowanych kobiet i mężczyzn, są zmodyfikowane przez programy do edycji grafiki, co może tworzyć mylny obraz rzeczywistości u ludzi, w szczególności osób dorastających. Aby dostosować się do wytworzonych sztucznych realiów postępują według zalecanych metod i czynności, nie zawsze służących zdrowiu, mających doprowadzić do osiągnięcia wzorcowej sylwetki.

Postęp technologiczny posiada jednak pozytywne aspekty. Pojawiły się różnego rodzaju narzędzia, takie jak aplikacje mobilne i platformy internetowe, pozwalające na monitorowanie czy śledzenie zmian nastroju. Taki sposób monitorowania pozwala na wygodne, niewymagające znaczącej fatygi, szybsze zapisywanie informacji o użytkowniku. Zaburzenia odżywiania często rozwijają się powoli, a pierwsze ich sygnały są trudne do zauważenia, bywają niezauważone[5]. Mając na uwadze fakt, iż jednym z kluczowych elementów skutecznego leczenia zaburzeń odżywiania jest wczesne wykrycie problemu.

---

[1] D. Devoe, A. Han, A. Anderson, D. K. Katzman, S. B. Patten, A. Soumbasis, J. Flanagan, G. Paslakis, E. Vyver, G. Marcoux, G. Dimitropoulos, *The impact of the COVID -19 pandemic on eating disorders: A systematic review*, „International Journal of Eating Disorders", nr. 1, 2022, s. 8-13.

[2] A. E. van Eeden, D. van Hoeken, H. W. Hoek, *Incidence, prevalence and mortality of anorexia nervosa and bulimia nervosa*, „Current Opinion in Psychiatry", nr. 1, 2021, s. 17-18.

[3] J. Wycisk, B. Ziółkowska, *Młodzież przeciwko sobie Zaburzenia odżywiania i samouszkodzenia – jak pomóc nastolatkom w szkole*, Wydawnictwo Difin, Warszawa 2010, s. 60-62.

[4] F. Figueiredo, J.M. Almeida F. Benevenuto K.P. Gummadi, *Does content determine information popularity in social media?: a case study of youtube videos' content and their popularity*, „ SIGCHI Conference on Human Factors in Computing Systems", nr. 4, 2014, s. 1.

[5] K. Gaber, B. Kuk, *Kiedy cierpisz na zaburzenia odżywiania*, Wydawnictwo Lekarskie PZWL, Warszawa 2022, s. 18.

Rozpoznanie objawów zaburzeń ma znaczny wpływ na skuteczniejsze wsparcie potencjalnego pacjenta w procesie leczenia w kwestii zaburzeń odżywiania.

Celem pracy jest opracowanie aplikacji do monitorowania nastroju z elementami wczesnego ostrzegania przed zaburzeniami odżywiania. Zamiarem pracy jest zwiększenie świadomości społecznej na temat narastającego problemu społecznego związanego z zaburzeniami odżywiania poprzez przygotowanie autorskiej aplikacji mobilnej ułatwiającej wykrycie problemu powstawania zaburzeń odżywiania. Aplikacja mobilna będzie umożliwiać użytkownikowi śledzenie swoich nastrojów, a na podstawie zebranych danych sygnalizować możliwe zaburzenia odżywiania. Wczesne wykrywanie problemu pozwoli na szybszą interwencję za pomocą skutecznych metod leczenia zaburzeń odżywiania.

Część teoretyczna zawiera wybrane zagadnienia wyjaśniające i opisujące zaburzenia odżywiania, zaburzenia karmienia, a także kluczową różnicę pomiędzy nimi oraz najważniejsze ich cechy. Opisano dwa sposoby klasyfikacji zaburzeń odżywiania zdefiniowane przez Światową Organizację Zdrowia oraz Amerykańskie Towarzystwo Psychiatryczne. Zostały również wymienione i opisane patologie i choroby związane z nieprawidłowym odżywianiem.

W części praktycznej uwaga została skupiona na opracowaniu aplikacji mobilnej Mood Tracker z elementami wczesnego ostrzegania przed zaburzeniami odżywiania. Aplikacja ta będzie pozwalać użytkownikom na dokonywanie zapisów odnośnie do swoich nastrojów z danego dnia oraz innych zachowań związanych z jedzeniem. Poprzez analizę danych, sukcesywnie gromadzonych przez aplikację, będzie możliwe dalsze wczesne ostrzeganie wynikające z potencjalnych objawów zaburzeń zgłoszonych przez użytkownika aplikacji. Aplikacja będzie stanowiła jedynie narzędzie do wczesnej sygnalizacji, uświadomienia możliwie występujących patologii żywieniowych. Kryteria informowania oparte zostały o publikacje z dziedziny dietetyki oraz psychodietetyki. Program w głównym założeniu nie będzie substytutem kontaktu z lekarzem bądź wiedzy lekarskiej. Sama aplikacja ma jedynie stanowić narzędzie mające na celu uświadomienie klienta aplikacji o potencjalnym problemie w postaci nieinwazyjnych ostrzeżeń, udostępnienia odpowiednich zaleceń i wsparcia dla użytkownika.

Pracę przygotowano w oparciu o literaturę przedmiotu oraz materiały źródłowe udostępnione przez Światową Organizację Zdrowia (WHO), Amerykańskie Towarzystwo Psychiatryczne, Narodowe Instytuty Zdrowia.

ROZDZIAŁ I

ZABURZENIA ODŻYWIANIA I MECHANIZMY PSYCHOLOGICZNE

W tej części pracy przedstawiony zostanie temat związany z zaburzeniami odżywiania, przyczyny ich powstawania, kategoryzacja, najważniejsze kryteria pozwalające na jednoznaczną klasyfikacje oraz objaśnienie terminu automonitoringu. Przedstawienie i wyjaśnienie tego tematu ma za zadanie uświadomić czytelnikowi istotę i kompleksowość problematyki zaburzeń odżywiania oraz wpływu, jaki może mieć systematyczna obserwacja własnych zachowań na proces zdrowienia.

1.1. Zaburzenia odżywiania

Przez termin zaburzenia odżywiania, określanym też zamiennie patologią jedzenia, chorobami dietetycznymi lub trudnościami żywieniowymi, rozumie się niezgodne, niezdrowe zachowania żywieniowe lub zachowania mające na celu monitorowanie wagi, regulacje wagi pod postacią nadmiernej kontroli masy ciała lub utratę wagi[6].

Konkretne zaburzenie lub jego fundamentalne wyznaczniki, pozwalające na precyzyjne zdiagnozowanie danej patologii żywieniowej, precyzyjnie zdefiniowane przez klasyfikacje chorobowe (np. wymuszanie wymiotów, kompensacje, mylny obraz własnej sylwetki, jej wielkości lub wyglądu ciała) mogą w dłuższym odstępie czasowym doprowadzić do pogorszenia stanu zdrowia fizycznego. Owe zaburzenia, pomimo swojej zależności od prezencji fizycznej, są głęboko zakorzenione na podłożu psychicznym. W długoterminowym ujęciu mogą mieć dotkliwe, negatywne skutki, wpływające na funkcjonowanie psychospołeczne osoby cierpiącej na nie. Rozpatrując zaburzenia odżywiania, należy mieć na uwadze, że nieuporządkowane zachowanie nie powinno być wtórne do choroby psychicznej[7].

Zaburzenia karmienia i odżywiania (*Feeding and Eating Disorder*), dotyczą niezgodnych z przyjętymi normami zachowań, związanych z aktem jedzenia i chociaż mają pewne wspólne cechy psychopatologiczne behawioralne – są to różne pojęcia i znacząco odmienne. W zaburzeniach karmienia (*feeding disorder*) nie występuje przejęcie własną masą i kształtem ciała, odnoszą się one do nieprawidłowych nawyków związanych z przyjmowaniem pokarmu[8]. Natomiast zaburzeniom odżywiania (*eating disorder*) towarzyszy nadmierne

[6] C.G. Fairburn, *Terapia poznawczo-behawioralna i zaburzenia odżywiania*, Wydawnictwo Uniwersytetu Jagielońskiego, Kraków 2013, s. 113.
[7] A. Brytek-Matera , *Zaburzenia odżywiania,* Wydawnictwo Lekarskie PZWL, Warszawa 2022, s. 5.
[8] A. Brytek-Matera, *Zaburzenia* ..., op. cit., s. 7.

zaabsorbowanie jedzeniem oraz masą, kształtem ciała[9]. Obecne staje się głębokie osadzenie poczucia własnej wartości, w dużym stopniu lub wyłącznie, w ocenie własnego wyglądu[10]. Osoba cierpiąca na zaburzenia odżywiania posiada zaburzoną percepcję własnego wizerunku (*body image*). Wizerunek ciała rozumiany jako obraz własnego ciała, który powstaje w umyśle[11].

Przyczyna powstania zaburzeń odżywiania jest niejednoznaczna, może występować z przyczyn interakcji czynników biologicznych, psychologicznych, rozwojowych i społeczno-kulturowych[12]:

1. **Biologiczne**

a) **Genetyka:** Narodowość nieazjatycka, powikłania w okresie ciąży[13]. Znaczna większość osób chorujących na anoreksję to kobiety[14]. Anoreksja i bulimia znacznie częściej występują u kobiet niż u mężczyzn. Około 90% osób z zaburzeniami odżywiania to kobiety, mężczyźni stanowią około 10%[15].

W ciągu życia na jadłowstręt psychiczny choruje 1 na 200 dziewcząt i kobiet oraz 1 na 2000 chłopców i mężczyzn[16]. Badania koncentrujące się na identyfikacji różnic w predykatorach rozwoju zaburzenia z objadaniem się u dzieci i młodzieży oraz późniejszego wieku zachorowania wskazują, iż istotnym czynnikiem ryzyka rozwinięcia BED w okresie adolescencji jest płeć żeńska[17].

b) **Neurobiologia:** Serotonina odgrywa relewantną funkcję w regulacji apetytu i samopoczucia. Serotonina może pośrednio lub bezpośrednio wpływać na rozwój zaburzeń odżywiania.

[9] A. Brytek-Matera, *Zaburzenia* ..., op. cit., s. 7.
[10] C.G. Fairburn, *Terapia poznawczo-behawioralna…,* s. 113.
[11] A. Brytek-Matera*, Obraz ciała - obraz siebie, Wizerunek własnego ciała w ujęciu społecznym*, Wydawnictwo Difin, Warszawa 2008, s.11.
[12] A.A. Rikani, Z. Choudhry, A.M. Choudhry, H. Ikram, M.W. Asghar, D. Kajal, A. Waheed, N. J. Mobassarah, *A critique of the literature on etiology of eating disorders*, „Annals of Neurosciences", 20, 2013, s. 158-160.
[13] A. Brytek-Matera, *Zaburzenia ..., op. cit., s. 56.*
[14] K. Gaber, B. Kuk, *Kiedy …*, s. 31.
[15] A. Brytek-Matera, *Zaburzenia* ..., op. cit., s. 35.
[16] A. Brytek-Matera, *Zaburzenia* ..., op. cit., s. 35.
[17] K. Jowik, A. Dutkiewicz , A. Słopień, M. Tyszkiewicz-Nwafor, *A multi-perspective analysis of dissemination, etiology, clinical view and therapeutic approach for binge eating disorder (BED)*, „Psychiatria Polska", 54, 2020, *s. 6.*

2. **Psychologiczne:** Perfekcjonizm, który może być wynikiem oczekiwań ze strony rodziców, którzy od małego dziecka oczekiwali jak najwięcej[18], impulsywność[19], poszukiwania novum, idee fixe, neurotyczność, trudności w kontaktach z rówieśnikami[20],
3. **Rozwojowe:** Przemoc emocjonalna w dzieciństwie, traumy powstałe w okresie adolescencji, bycie ofiarą gwałtu lub napaści seksualnej – są to czynniki często spotykane z zaburzeniami odżywiania[21],
4. **Społeczno-kulturowe –** Zachodnia kultura i użytkowanie portali społecznościowych, współczesne wzorce piękna, obraz ciała, który chce się uzyskać. Wzrost trendu stron pro-ana, które według definicji promują ideologię głoszącą specyficzne podejście do anoreksji jako indywidualnego stylu życia[22].

Problem polegający na trudności klasyfikacji zaburzeń odżywiania, wynika z tego, że ludzie z zaburzeniami odżywania, które nie spełniają dokładnie sprecyzowanych przez obie wyżej wymienione klasyfikacje kryteriów behawioralnych, wcześniej były one definiowane jako inaczej niesprecyzowane. Obecnie mogą być uznane jako: pozostałe określone zaburzenia odżywania i karmienia lub jako niesprecyzowane zaburzenia odżywiania w DSM-5 TR, wraz z wydaniem rewizji tekstu 5 edycji DSM (18 marca 2022r) pojawiło się doprecyzowanie. Aby zwiększyć klarowność i uniknąć błędu klasyfikacji, wydzielono rozgraniczenie między (typową) anoreksją psychiczną oraz atypową anoreksją psychiczną[23].

Ten sam problem występuje w klasyfikacji ICD-11, gdzie takie zaburzenia jak[24]:

- atypowa anoreksja (*atypical anorexia nervosa*),
- podprogowa bulimia (*subthreshold bulimia nervosa*),
- zaburzenia z przeczyszczaniem się (*purging disorder*),

[18] K. Gaber, B. Kuk, *Kiedy* …, s. 64
[19] Z. He, W. Yang, *Impulsiveness as potential moderators of the relation between social media dependence and eating disorders risk, „BMC Psychology", 10, 2022, s. 2-3.*
[20] D. Mroczkowska, B. Ziółkowska, A. Cwojdzińska, *Zaburzenia odżywiania poradnik dla rodziców i bliskich*, Wydawnictwo Naukowe Scholar, Warszawa 2007, s. 19.
[21] A. Malet-Karas, D. Bernard-Wallendorf, E. Piet, E. Bertin, *Eating disorders as a repercussion of sexual assault, a consequence to consider, https://assets.researchsquare.com/files/rs-909393/v1/e7ba13e5-4cb4-4a00-ac6a-63fa9e07dfaa.pdf?c=1632178445, dostęp: 05.02.2023.*
[22] K. Gaber, B. Kuk, *Kiedy* …, opt. cit, s. 83
[23] Opracowanie na podstawie: Zasób portalu Psychiatry, https://www.psychiatry.org/File%20Library/Psychiatrists/Practice/DSM/DSM-5-TR/APA-DSM5TR-OtherSpecifiedFeedingDisorder.pdf, dostęp: 05.02.2023.
[24] P. Hay, Current approach to eating disorders: a clinical update, „Internal Medicine Journal", nr. 50, 2020, s. 25.

- zespół nocnego podjadania (*night eating syndrome*).

Zostały sklasyfikowane jako pozostałe określone zaburzenia karmienia i odżywiania. Takie zaburzenia jak alkoreksja są stosunkowo nowymi zaburzeniami żywieniowymi, które dotąd nie zostały ujęte w klasyfikacji chorób ICD-11.

1.2. Klasyfikacja zaburzeń odżywiania

18 maja 2013 roku została opublikowana 5 edycja klasyfikacji zaburzeń psychicznych, opracowana przez Amerykańskie Towarzystwo Psychiatryczne ,,Diagnostic and Statistical Manual of Mental Disorders''. Opisana w poniższej tabeli (patrz. Tabela 1. Klasyfikacja zaburzeń odżywiania według DSM-5), klasyfikacja uznawana jest jako główne narzędzie i autorytet do określania diagnoz w dziedzinie psychiatrii w Stanach Zjednoczonych. Najnowszą edycją klasyfikacji jest DSM-5 TR ,,Diagnostic and Statistical Manual of Mental Disorders 5th edition text revision'' wprowadzającą poprawki do oryginału.

Tabela 1. Klasyfikacja zaburzeń odżywiania według DSM-5

Pica (zwana łaknieniem spaczonym).
Zaburzenie ruminacji
Zaburzenia unikania/restrykcyjnego przyjmowania pokarmu
Anorexia nervosa
Bulimia
Zaburzenie napadowego objadania się
Inne określone zaburzenie jedzenia i odżywiania się
Nieokreślone zaburzenie jedzenia i odżywiania się

Źródło: Opracowanie na podstawie: C.G. Fairburn, *Terapia poznawczo-behawioralna i zaburzenia odżywiania*, Wydawnictwo Uniwersytetu Jagielońskiego, Kraków 2013, s. 66.

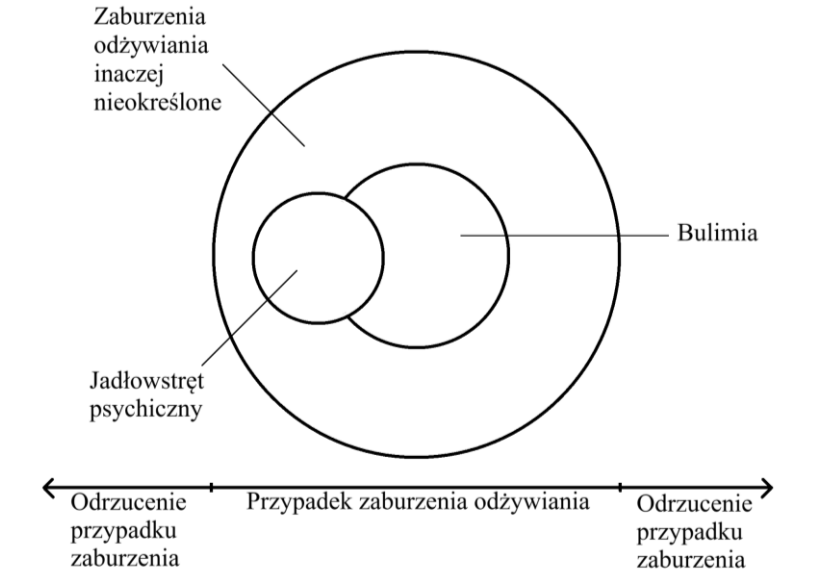
Warto podkreślić kluczowe różnice między aktualną wersją, a jej poprzedniczką, ponieważ w powszechnie dostępnych źródłach internetowych przestarzała klasyfikacja DSM-IV TR wciąż określana jest jako podstawowe źródło wiedzy z dziedziny patologii odżywiania. W odniesieniu do swojej poprzedniczki zmieniła terminologię „Zaburzenia odżywiania" (użytej w poprzedniej edycji oznaczonej DSM-IV TR) na „Zaburzenia karmienia i odżywiania" i zawiera kluczowe zmiany w stosunku do poprzednika DSM-IV, modyfikacje

obejmują rozdzielenie i doprecyzowanie zaburzeń odżywiania inaczej nieokreślonych (*Eating Disorders Not Other Specified*), na między innymi[25]:

- zaburzenia unikania/restrykcyjnego przyjmowania pokarmu,
- zaburzenia ruminacji (zwanymi zaburzeniami przeżuwania),
- pica (zwana łaknieniem spaczonym).

Przyczyną rozszerzenia rozdziału zaburzeń karmienia i odżywiania w DSM V było zaniedbanie o charakterze nozologicznym, które uznawało najczęściej występującą kategorię jako kategorię "szczątkową" (~60%)[26] i jednocześnie najbardziej niedoprecyzowaną w diagnozie. Na ilustracji (patrz Rysunek 1. Kryteria diagnostyczne zaburzeń odżywiania według DSM IV), przedstawiono dwa nachodzące na siebie wewnętrzne okręgi przedstawiające odpowiednio: jadłowstręt psychiczny (mniejszy okrąg), bulimia (większy okrąg). Przedstawiają one ówcześnie przyjętą kolejność, gdzie pierwszeństwo diagnozy miał jadłowstręt psychiczny, następnie bulimia, a ostatnie w kolejności są określane jako granica odstępstwa od normy, według ówczesnej terminologii nazwane zaburzenia odżywiania inaczej nieokreślone.

Rysunek 1. Kryteria diagnostyczne zaburzeń odżywiania według DSM IV



Źródło: C.G. Fairburn, K Bohn, *Eating disorder NOS (EDNOS): an example of the troublesome "not otherwise specified" (NOS) category in DSM-IV*, ,,Behaviour Research and Therapy'', nr. 43, 2005, s. 693.

[25] *Opracowanie na podstawie: C.G. Fairburn, K. Bohn, Eating disorder NOS (EDNOS): an example of the troublesome "not otherwise specified" (NOS) category in DSM-IV, ,,Behaviour Research and Therapy'', nr. 43, 2005, s. 693-697.*

[26] Opracowanie na podstawie: C.G. Fairburn, K. Bohn, *Eating disorder NOS (EDNOS): an example of the troublesome "not otherwise specified" (NOS) category in DSM-IV*, ,,Behaviour Research and Therapy'', nr. 43, 2005, s. 693.

Dokonano zmian zarówno w nielicznych determinantach diagnostycznych konkretnych zaburzeń odżywiania np.: usunięcie wtórnego braku miesiączki u kobiet, które przeszły okres dojrzewania w diagnozie jadłowstrętu psychicznego (*anorexia nervosa*). Powodem było, że u pacjentek, u których brak miesiączki nie występował, udokumentowano podobne objawy[27].

International Classification of Diseases (ICD) - międzynarodowa klasyfikacja chorób i procedur medycznych, stworzona przez Światową Organizację Zdrowia (WHO), służy szerokiemu zakresowi zastosowań na całym świecie i zapewnia krytyczną wiedzę na temat zakresu, przyczyn i konsekwencji chorób i śmierci ludzi na całym świecie za pośrednictwem danych, które są zgłaszane i kodowane za pomocą ICD.

Z dniem 1 stycznia 2022 roku ukazała się 11 edycja przyjęta do użycia w praktyce. Opisana w poniższej tabeli (patrz Tabela 2. Klasyfikacja zaburzeń odżywiania według ICD), klasyfikacja uznaje zaburzenia odżywiania i karmienia jako zaburzenia z rodziny zaburzeń psychicznych, behawioralnych[28].

Tabela 2. Klasyfikacja zaburzeń odżywiania według ICD

Jadłowstręt psychiczny
Bulimia psychiczna
Zaburzenie z napadami objadania się
Zaburzenie z ograniczeniem/unikaniem przyjmowania pokarmów
Pica (zwana łaknieniem spaczonym).
Zaburzenia przeżuwania
Pozostałe określone zaburzenia karmienia i odżywiania
Niesprecyzowane określone zaburzenia karmienia i odżywiania

Źródło: Opracowanie na podstawie: Zasoby portalu World Health Organisation, https://icd.who.int/browse11/l-m/en#/http%3a%2f%2fid.who.int%2ficd%2fentity%2f1412387537, dostęp: 05.02.2023

Dziedzina wiedzy zajmująca się zaburzeniami jest w stałym procesie rozwoju, z tego powodu, w autorskiej aplikacji Mood Tracker, przyjęto uproszczenia wybierając najbardziej reprezentatywne i charakterystyczne rodzaje zaburzeń do celów diagnozy. Spośród przedstawionych kategorii uproszczeniu uległy, "Inne" i "Nieokreślone" w klasyfikacji DSM (zobacz Tabela 1. Klasyfikacja zaburzeń odżywiania

[27] C.G. Fairburn, *Terapia poznawczo-behawioralna …*, *op. Cit. s.9*
[28] Źródło: Opracowanie na podstawie: Zasoby portalu World Health Organisation, https://www.who.int/standards/classifications/classification-of-diseases, dostęp: 05.02.2023

według DSM-5), oraz "Pozostałe" i "Niesprecyzowane" w klasyfikacji ICD (zobacz Tabela 2. Klasyfikacja zaburzeń odżywiania według ICD).

1.3. Główne zaburzenia odżywiania

Anoreksja jest związana z pokaźnie niewysoką masą ciała w odniesieniu do wzrostu, wieku, stadium rozwoju lub historii masy ciała oraz skrajnymi postawami i zachowaniami, które odróżniają ją od zwyczajnej diety i "normatywnego rozczarowania" z postaci i wagi ciała[29].

Tabela 3. Anoreksja - kryteria diagnostyczne

1	Wskaźnik BMI niżej niż 18,5 kg/m2. Zauważalnie obniżona waga przyrównując do wzrostu ciała, wieku, etapu życia lub nienaturalnie niska waga dla osoby cierpiącej w odniesieniu do jej wcześniejszego wizerunku.
2	Zmniejszenie podaży kalorycznej dla pacjenta nie może być spowodowane brakiem pożywienia lub powikłaniami chorobowymi
3	Niskiej masie ciała towarzyszą zachowania zapobiegające przywróceniu prawidłowej masy ciała, które mogą obejmować zachowania mające na celu ograniczanie jedzenia, przeczyszczanie organizmu (środkami przeczyszczającymi, prowokowaniem wymiotów). Zachowania te zazwyczaj związane są z lękiem przed przyrostem masy ciała
4	Zaburzoną percepcję własnego wizerunku

Źródło: Opracowanie na podstawie: Zasoby portalu World Health Organisation, https://icd.who.int/browse11/l-m/en#/http://id.who.int/icd/entity/263852475, dostęp: 05.02.2023

W powyższej tabeli (patrz. Tabela 3. Anoreksja - kryteria diagnostyczne), zaprezentowano kryteria diagnostyczne, opracowane jako wskaźniki wykorzystywane do wczesnego ostrzegania przed jadłowstrętem psychicznym w autorskiej aplikacji Mood Tracker.

Istnieje odmiana jadłowstrętu w przypadku, którego głównym czynnikiem rozgraniczającym te definicje jest fakt, że osoby z atypową (nazywaną zamiennie nietypową) anoreksją, mogą mieć normalną wagę, nadwagę, otyłość lub lekką niedowagę, co zachodzi na skutek obniżenia podaży kalorycznej.

[29] Opracowanie na podstawie: Zasoby portalu World Health Organisation, https://icd.who.int/browse11/l-m/en#/http://id.who.int/icd/entity/263852475, dostęp: 05.02.2023

Zaburzenie z ograniczeniem/unikaniem przyjmowania pokarmów (*Avoidant-Restrictive Food Intake Disorder*) i jadłowstrętem psychicznym (*Anorexia Nervosa*) dzielą ze sobą podobieństwa. Oba wyżej wymienione zaburzenia zawierają w sobie restrykcyjne nawyki żywieniowe. Zaburzenie z ograniczeniem/unikaniem przyjmowania pokarmów charakteryzuje[30] się brakiem zainteresowania jedzeniem, na skutek poprzednich negatywnych doświadczeń, natomiast objawem głównym anoreksji jest sytuacja, w której osoba wychudzona widzi siebie w lustrze grubszą, niż rzeczywiście jest[31].

Tabela 4. Zaburzenie z ograniczeniem/unikaniem przyjmowania pokarmów – kryteria diagnostyczne

1	Ograniczanie spożycia pokarmu skutkuje występowaniem minimum jednej z wymienionych sytuacji. Spożycie niewystarczającej ilości lub różnorodności żywności w celu zaspokojenia odpowiedniego zapotrzebowania na energię lub składniki odżywcze, które spowodowało znaczną utratę wagi, klinicznie istotne niedobory żywieniowe, uzależnienie od doustnych suplementów diety lub karmienia przez zgłębnik lub w inny sposób negatywnie wpłynęło na zdrowie fizyczne danej osoby. Znaczne upośledzenie w osobistych, rodzinnych, społecznych, edukacyjnych, zawodowych lub innych ważnych obszarach funkcjonowania (np. z powodu unikania lub stresu związanego z uczestnictwem w doświadczeniach społecznych związanych z jedzeniem).
2	Wzorzec zachowań żywieniowych nie jest motywowany zaabsorbowaniem masą ciała lub kształtem.
3	Ograniczone spożycie pokarmu i wynikająca z tego utrata masy ciała (lub brak przybierania na wadze) lub inny wpływ na zdrowie fizyczne lub związane z tym upośledzenie funkcjonalne, nie są spowodowane niedostępnością żywności; nie są przejawem innego schorzenia (np. alergii pokarmowych, nadczynności tarczycy) lub zaburzeń psychicznych i nie są spowodowane działaniem substancji lub leku, w tym skutkami odstawienia.

Źródło: Opracowanie na podstawie: Zasoby portalu World Health Organisation, https://icd.who.int/browse11/l-m/en#/http%3a%2f%2fid.who.int%2ficd%2fentity%2f1242188600, dostęp: 05.02.2023

[30] Opracowanie na podstawie: Zasoby portalu World Health Organisation, https://icd.who.int/browse11/l-m/en#/http%3a%2f%2fid.who.int%2ficd%2fentity%2f1242188600, dostęp: 05.02.2023
[31] K. Gaber, B. Kuk, *Kiedy* …, s. 20

W powyższej tabeli (patrz Tabela 4. Zaburzenie z ograniczeniem/unikaniem przyjmowania pokarmów – kryteria diagnostyczne), zaprezentowano kryteria diagnostyczne używane jako wskaźniki detekcji powyższego zaburzenia odżywiania w autorskiej aplikacji.

Diagnoza Zaburzenia z ograniczeniem/unikaniem przyjmowania pokarmów (*Avoidant-Restrictive Food Intake Disorder*) i jadłowstrętu psychicznego (*Anorexia Nervosa*) wykluczają się[32].

Jadłowstręt psychiczny i bulimia to dwa skrajnie odmienne zaburzenia odżywiania, które posiadają podobieństwo dla obu, charakteryzujące się internalizacją szczupłych, idealnych i ekstremalnych zachowań kontrolnych wagi. W obu przypadkach przewartościowanie wagi i kształtu ciała[33]. Osoby cierpiące na anoreksję i bulimię psychiczną charakteryzują się większymi trudnościami w identyfikowaniu, rozpoznawaniu i nazywaniu własnych emocji oraz bodźców[34]. Wspólną cechą obu zaburzeń jest występowanie wśród pacjentów czynników psychologicznych takich jak: wrażliwość na krytycyzm, deficytów poznawczych, zobojętnienie emocjonalne bądź chwiejność uczuciowa.

Tabela 5. Bulimia psychiczna - kryteria diagnostyczne

1	Częste, nawracające epizody objadania się (np. raz w tygodniu bądź częściej przez okres przynajmniej 1 miesiąca). Objadanie się precyzuje się jako dyskretny okres (np. 2 godziny), w trakcie, którego osoba odczuwa utratę kontroli nad własnymi zachowaniami żywieniowymi i je pokaźnie więcej w porównaniu niż zwykle.
2	Powtarzające się działania kompensacyjne, celem których jest zapobieganie przybraniu na wadze (np. raz w tygodniu lub częściej przez okres co najmniej 1 miesiąca). Najczęstszym zachowaniem kompensacyjnym są samoczynne wymioty, które pojawiają się w ciągu godziny od epizodu objadania się.
3	Zaabsorbowanie wagą lub wyglądem, jeśli nie zostało wyraźnie zgłoszone, może objawiać się takimi zachowaniami jak wielorazowe sprawdzanie masy ciała; sprawdzanie obwodów ciała za pomocą taśmy mierniczej lub odbicia w lustrach; systematyczne monitorowanie ilości kalorii w żywności lub poszukiwania informacji o tym, jak schudnąć; lub przez radykalne zachowania unikające, takie jak

[32] Źródło: Opracowanie na podstawie: Zasoby portalu World Health Organisation, https://icd.who.int/browse11/l-m/en#/http%3a%2f%2fid.who.int%2ficd%2fentity%2f1242188600, dostęp: 05.02.2023

[33] P. Hay, *Current approach to eating disorders: a clinical update*, „Internal Medicine Journal", nr. 50, 2020, s. 24-26.

[34] A. Brytek-Matera, *Zaburzenia* ..., op. cit., s. 29.

	negowanie posiadania luster w domu, unikanie opiętych ubrań lub odmowa poznania własnej wagi lub zakupu odzieży o określonym rozmiarze.
4	Upośledzenie w zakresie osobistych, społecznych, edukacyjnych, zawodowych lub innych ważnych obszarach funkcjonowania.
5	Objawy nie spełniają wymagań diagnostycznych dla jadłowstrętu psychicznego.

Źródło: Opracowanie na podstawie: Zasoby portalu World Health Organisation, https://icd.who.int/browse11/l-m/en#/http%3a%2f%2fid.who.int%2ficd%2fentity%2f509381842, dostęp: 05.02.2023

W powyższej tabeli (patrz Tabela 5. Bulimia psychiczna - kryteria diagnostyczne), przedstawiono kryteria kliniczne użyte jako wytyczne w autorskiej aplikacji do wczesnego rozpoznawania bulimii.

Rzadkie przejadanie się w trakcie kulturowo usankcjonowanych świąt, bądź okazjonalnych uroczystości nie należy określać jako objadanie się w celu przypisania diagnozy bulimii. Ćwiczenia kwalifikują się jako niewłaściwe zachowanie kompensacyjne jedynie wtedy, kiedy są wyjątkowo intensywne lub długotrwałe, bądź są wykonywane z wyłączeniem innych czynności lub mimo wyczerpania, bólu lub uszkodzenia ciała[35].

Amerykańskie Towarzystwo Psychiatryczne do momentu publikacji piątej edycji klasyfikacji zaburzeń psychicznych, na skutek niedoprecyzowania uznawało bulimie oraz zaburzenie z napadami objadania jako zaburzenie bardzo często tożsame lub trafiało do zaburzeń odżywiania inaczej niesprecyzowanych (patrz Rysunek 1. Kryteria diagnostyczne zaburzeń odżywiania według DSM IV).

Tabela 6. Zaburzenie z napadami objadania- kryteria diagnostyczne

1	Występują epizody objadania, które pojawiają się stosunkowo często i są nawracające. Napad objadania to okres, w którym dana osoba traci kontrolę nad swoimi zachowaniami żywieniowymi, zmieniając je drastycznie lub znacząco, zwiększa ilość podaży energii w postaci pożywienia. Po rozpoczęciu jedzenia pojawiają się trudności z jego zaprzestaniem, określane przez pacjentów jako utrata kontroli. W momencie objadania się pacjent świadomy może być tego, że zaprzestanie starań ukierunkowanych na kontrolę jedzenia jest bezskuteczne, ponieważ skończy się to silniejszą kompensacją (ćwiczeniami fizycznymi, zmniejszeniem podaży kalorii).

[35] Opracowanie na podstawie: Zasoby portalu World Health Organisation, https://icd.who.int/browse11/l-m/en#/http%3a%2f%2fid.who.int%2ficd%2fentity%2f509381842dostęp: 05.02.2023

2	W okresie, gdy występuje objadanie się może towarzyszyć przesadne zachowanie dążące do zatrzymania wzrostu masy ciała.
3	Objawy i zachowania nie są spowodowane wpływem innych spożywanych substancji lub leków na ośrodkowy układ nerwowy, ani efektami ich odstawienia.
4	Upośledzeniem w relacjach osobistych, rodzinnych, społecznych, edukacyjnych, zawodowych lub pozostałych obszarach funkcjonowania.

Źródło: Opracowanie na podstawie: Zasoby portalu World Health Organisation, https://icd.who.int/browse11/l-m/en#/http%3a%2f%2fid.who.int%2ficd%2fentity%2f1673294767, dostęp: 05.02.2023

Powyższa tabela (patrz Tabela 6. Zaburzenie z napadami objadania - kryteria diagnostyczne), prezentuje kryteria kliniczne, wykorzystane jako wskazówki, w autorskiej aplikacji do wczesnego ostrzegania przed zaburzeniem z napadami objadania.

Nieczęste objadanie się podczas świąt nie powinno być określane jako powód do zdiagnozowania zaburzenia objadania się. Osoby, które zgłaszają wzorce przejadania się, które nie spełniają definicji objadania się, nie powinny być diagnozowane zaburzeniem z napadami objadania się. Przykłady obejmują bezmyślne jedzenie, któremu można się oprzeć lub które można zatrzymać (np. jeśli występuje rozproszenie lub przerwanie) lub jedzenie więcej niż pierwotnie zamierzano bez poczucia utraty kontroli, nawet jeśli ten rodzaj jedzenia jest niepokojący[36].

W poniższej tabeli (patrz Tabela 7. Zaburzenie z napadami objadania i uzależnienie od jedzenia - różnice), przedstawiono różnice między zaburzeniem z napadami objadania się a uzależnieniem od pożywienia pod względem różnych wskaźników. Zarówno zaburzenie z napadami objadania się, jak i uzależnienie od jedzenia, charakteryzują się utratą kontroli nad konsumpcją, ciągłym nadużywaniem pomimo negatywnych konsekwencji i powtarzającymi się nieudanymi próbami ograniczenia konsumpcji[37].

Tabela 7. Zaburzenie z napadami objadania i uzależnienie od jedzenia - różnice

Wskaźnik	**Zaburzenie z napadami objadania się**	**Uzależnienie od pożywienia**
Występowanie	Epizody zaburzonego zachowania	Stałe

[36] Tłumaczenie własne, *Boundary with Normality (Threshold)*, https://icd.who.int/browse11/l-m/en#/http%3a%2f%2fid.who.int%2ficd%2fentity%2f509381842

[37] E. Giacomo, F. Aliberti, F. Pescatore, M. Santorelli, R. Pessina, V. Placenti, F. Colmegna, M. Clerici, *Disentangling binge eating disorder and food addiction: a systematic review and meta-analysis*, ,,Eating and Weight Disorders - Studies on Anorexia, Bulimia and Obesity", nr. 27, 2022, 1964

Poczucie głodu	Spożywanie jedzenia bez uczucia głodu	Głód określonego pożywienia
Smak	Nieistotny	Pragnienie pożywienia subiektywnie określanego „smacznym”
Przyjemność	Brak	Występuje
Funkcja jedzenia	Redukcja napięcia	Wywołanie satysfakcji
Towarzystwo	Samotność	Nie ma znaczenia
Poczucie winy	Występuje	Brak

Źródło: Opracowanie na podstawie: M. Bąk-Sosnowska, *Differential criteria for binge eating disorder and food addictionin the context of causes and treatment of obesity*, „Psychiatria Polska”, nr. 51, 2017, s. 253-254.

Zaburzenie z napadami objadania i zespół ruminacji mają podobieństwa, powiązane tym, że oba dotyczą powtarzających się i problematycznych wzorców jedzenia i spożywania żywności. BED charakteryzuje się częstymi epizodami spożywania dużych ilości jedzenia w krótkim okresie, często z towarzyszącym uczuciem utraty kontroli. Zespół ruminacji to powtarzane wymioty i ponowne rozgryzanie jedzenia, które zostało częściowo strawione[38].

Tabela 8. Zaburzenia przeżuwania – kryteria diagnostyczne

1	Celowe i powtarzające się przywoływanie wcześniej połkniętego pokarmu z powrotem do ust (tj. niedomykalność), które mogą być ponownie żute i ponownie połykane (tj. przeżuwanie) lub może być celowo wypluwane (ale nie jak w wymiotach).
2	Zachowania niedomykalności są częste (co najmniej kilka razy w tygodniu) i utrzymują się przez okres co najmniej kilku tygodni.
3	Diagnoza powinna być przypisana tylko osobom, które osiągnęły wiek rozwojowy co najmniej 2 lata.
4	Zachowanie niedomykalności nie jest przejawem innego stanu chorobowego, który bezpośrednio powoduje niedomykalność (np. zwężenia przełyku lub zaburzenia nerwowo-mięśniowe wpływające na funkcjonowanie przełyku) lub powoduje nudności lub wymioty (np. zwężenie odźwiernika).

Źródło: Opracowanie na podstawie: Zasoby portalu World Health Organisation, https://icd.who.int/browse11/l-m/en#/http%3a%2f%2fid.who.int%2ficd%2fentity%2f1205760590, dostęp: 05.02.2023

[38] Opracowanie na podstawie: Zasoby portalu World Health Organisation, https://icd.who.int/browse11/l-m/en#/http%3a%2f%2fid.who.int%2ficd%2fentity%2f1205760590, dostęp: 05.02.2023

Tabela powyżej (patrz. Tabela 8. Zaburzenia przeżuwania – kryteria diagnostyczne), opisuje wskazówki oceny stosowane w autorskiej aplikacji Mood Tracker do wczesnego rozpoznawania zaburzenia przeżuwania.

Zaburzenie przeżuwania-niedomykalności nie powinno być diagnozowane u niemowląt. Podobne zjawiska u niemowląt należy zdiagnozować jako zespół przeżuwania Niemowląt w grupie czynnościowe zaburzenia trawienia niemowląt, małych dzieci lub dzieci w rozdziale ICD-11 dotyczącym chorób układu pokarmowego. Zaburzenie przeżuwania-niedomykalności należy odróżnić od wymiotów wywołanych przez siebie, które mogą wystąpić w ramach prezentacji jadłowstrętu psychicznego lub bulimii. Wymioty wywołane przez siebie mogą również wystąpić jako kulturowo usankcjonowana praktyka (np. wśród praktykujących jogę), która nie jest związana z zaburzeniem psychicznym.

Granica z psychogennymi wymiotami, odróżnienie od "psychogennych wymiotów" lub wymiotów jako somatycznego wyrazu cierpienia, szczególnie w Azji Południowej, opiera się na fakcie, że niedomykalność w zaburzeniu przeżuwania-niedomykalności jest zazwyczaj wolicjonalna i celowa.

Zaburzenie przeżuwania-niedomykalności i Pica posiadają podobieństwa, ponieważ dotyczą spożywania przedmiotów nienależących do jedzenia. Pica to stan, w którym osoby mają trwałą potrzebę spożywania substancji nienutrientnych, takich jak brud, kreda, lód lub papier[39]. Z kolei zaburzenie przeżuwania-niedomykalności, oznacza powtarzające się wypluwanie i ponowne żucie pokarmu, który został częściowo strawiony[40].

Tabela 9. Pica (zwana łaknieniem spaczonym) - kryteria diagnostyczne

1	Regularne spożywanie substancji nieprzeznaczonych do spożycia, takich jak przedmioty i materiały nieżywnościowe (np. glina, gleba, kreda, gips, plastik, metal i papier) lub surowe składniki żywności (np. duże ilości soli lub mąki kukurydzianej).
2	Spożycie substancji nieprzeznaczonych do spożycia jest na tyle trwałe lub ciężkie, że wymaga pomocy klinicznej. Oznacza to, że zachowanie powoduje uszkodzenie

[39] D. Mroczkowska, B. Ziółkowska, A. Cwojdzińska, *Zaburzenia* …, s. 19.
[40] Opracowanie na podstawie: Zasoby portalu World Health Organisation, https://icd.who.int/browse11/l-m/en#/http%3a%2f%2fid.who.int%2ficd%2fentity%2f1205760590, dostęp: 05.02.2023

	lub znaczne ryzyko dla zdrowia lub upośledzenie funkcjonowania ze względu na częstotliwość, ilość lub charakter spożywanych substancji lub przedmiotów.
3	W oparciu o wiek i poziom funkcjonowania intelektualnego oczekuje się, że jednostka rozróżni substancje jadalne i niejadalne. W typowym rozwoju występuje to w wieku około 2 lat.
4	Objawy lub zachowania nie są przejawem innego stanu chorobowego (np. niedoboru żywieniowego).

Źródło: Opracowanie na podstawie: Zasoby portalu World Health Organisation, https://icd.who.int/browse11/l-m/en#/http%3a%2f%2fid.who.int%2ficd%2fentity%2f833390860, dostęp: 05.02.2023

Powyższa tabela (zobacz Tabela 9. Pica, inaczej zwana łaknieniem spaczonym - kryteria diagnostyczne), przedstawia kryteria diagnostyczne wykorzystane jako wskaźniki w autorskiej aplikacji do wczesnego identyfikowania łaknienia spaczonego.

Wiele kobiet w ciąży pragnie lub je substancje np. kreda lub lód. Ponadto spożywanie substancji jest kulturowo usankcjonowaną praktyką wśród niektórych grup. Rozpoznanie produktu Pica powinno być przypisane do takiego zachowania tylko wtedy, gdy jest ono trwałe lub potencjalnie wystarczająco niebezpieczne, aby wymagać szczególnej pomocy klinicznej[41]. Niemowlęta i bardzo małe dzieci wkładają do ust przedmioty nieżywnościowe jako środek eksploracji sensorycznej, diagnoza Pica nie powinna być stosowana do tego zjawiska.

Należy mieć na uwadze, że ponad 90% przypadków diagnozowanych jest przed 25 rokiem życia[42]. Grupa, która w sposób szczególny narażona jest na wystąpienie alkoreksji stanowią studenci[43]. Osoby z alkoreksją, w przeciwieństwie do osób z anoreksją, podejmują działania kompensacyjne tylko w sytuacjach, kiedy planują spożywać alkohol[44].

Tabela 10. Alkoreksja - kryteria diagnostyczne

1	Planowanie ograniczania pokarmu celem kompensacji kalorii zawartych w napojach alkoholowych.
2	Prowokowanie wymiotów bądź stosowanie środków przeczyszczających i/lub moczopędnych celem redukcji puli spożytych kalorii.

[41] Opracowanie na podstawie: Zasoby portalu World Health Organisation, https://icd.who.int/browse11/l-m/en#/http%3a%2f%2fid.who.int%2ficd%2fentity%2f833390860, dostęp: 05.02.2023
[42] A. Brytek-Matera, Zaburzenia ..., op. cit., s. 36.
[43] K. Szynal, M. Górski, M. Grajek, K. Ciechowska, R. Polaniak, Drunkorexia – knowledge review, „Psychiatria Polska", nr. 56 , 2022, s. 1132
[44] Opracowanie własne na podstawie: K. Szynal, M. Górski, M. Grajek, K. Ciechowska, R. Polaniak, *Drunkorexia – knowledge review*, „Psychiatria Polska", nr. 56 , 2022, s. 1132-1137

3	Nadmierne ćwiczenia fizyczne w celu zmniejszenia ilości kalorii pochodzących ze spożycia alkoholu.
4	Planowanie ograniczania spożycia żywności w celu zwiększenia odurzenia po spożyciu alkoholu

Źródło: Opracowanie własne na podstawie: K. Szynal, M. Górski, M. Grajek, K. Ciechowska, R. Polaniak, *Drunkorexia – knowledge review*, „Psychiatria Polska", nr. 56, 2022, s. 1-11

W powyższej tabeli (patrz Tabela 10. Alkoreksja - kryteria diagnostyczne) zostały ukazane kryteria diagnostyczne, które stanowią punkt odniesienia w autorskiej aplikacji do wczesnej detekcji alkoreksji.

1.4 Automonitoring

Automonitoring polega na zbieraniu danych przez pacjenta dotyczących jego zachowania związanego z jedzeniem, takich jak ilość zjedzonych posiłków, składniki odżywcze, a także emocje i myśli związane z jedzeniem. Jest to ważna część terapii leczenia zaburzeń odżywiania, ponieważ pozwala pacjentom na uświadomienie sobie swoich nawyków żywieniowych i wprowadzanie zmian w celu osiągnięcia zdrowego stylu życia. Automonitoring w czasie rzeczywistym to stała odbywająca się w „każdej chwili" rejestracja istotnych zachowań, myśli, odczuć, zdarzeń. Monitoring powinien zostać uruchomiony od samego początku terapii[45]. Powyższy mechanizm pozwala dostrzec zmiany i osiągnięcia, które często mogą zostać niezauważone.

Tabela 11. Automonitoring - główne zalety

Pomaga zrozumieć, co dokładnie dzieje się w danym momencie, z chwili na chwilę oraz z dnia na dzień.
Ułatwia wprowadzanie zmiany, nabranie większej świadomości

Źródło: Opracowanie na podstawie C.G. Fairburn, *Terapia poznawczo-behawioralna i zaburzenia odżywiania*, Wydawnictwo Uniwersytetu Jagielońskiego, Kraków 2013, s. 66.

W przedstawionej tabeli (patrz Tabela 11. Automonitoring - główne zalety), zawarte są informacje na temat głównych korzyści wynikających z automonitoringu, między innymi pomaga on zrozumieć, co dokładnie dzieje się w danym momencie, z chwili na chwilę oraz z dnia na dzień. Ponadto ułatwia wprowadzanie zmiany i nabieranie większej świadomości.

[45] C. G.Fairburn, Terapia poznawczo-behawioralna …, op. Cit. s. 66.

Tabela 12. Zaburzeń odżywiania - utrudnienia w zakresie leczenia

Brak szczerości ze strony pacjenta, jego mechanizmy obronne – brak odpowiedzialności co się dzieje w procesie.
Wstyd mogący blokować możliwość wykonywania automonitoringu, a ten z kolei jest niezbędny, by móc sprawdzać, jak idą postępy i czy napady się powtarzają, czy maleją. Lęk przed jedzeniem oraz byciem ocenianym

Źródło: Opracowanie na podstawie: K. Gaber, B. Kuk, Kiedy cierpisz na zaburzenia odżywiania, Wydawnictwo Lekarskie PZWL, Warszawa 2022, s. 120-123.

Powyższa tabela (patrz Tabela 12. Zaburzeń odżywiania - utrudnienia w zakresie leczenia), znajdują się informacje dotyczące trudności związanych z leczeniem zaburzeń żywieniowych. Pacjenci mogą nie być szczerymi, stosować mechanizmy obronne i unikać odpowiedzialności za to, co dzieje się w trakcie procesu terapeutycznego. Ponadto, uczucie wstydu może stanowić barierę dla prowadzenia automonitoringu, który jest istotny do oceny postępów i monitorowania.

Szeroko przedstawione problemy związane z zaburzeniami odżywiania, etiologia, klasyfikacja zaburzeń, podobieństwa i różnice pomiędzy niektórymi zaburzeniami odżywiania, miało na celu uświadomienie potencjalnym użytkownikom powagi i ważności problemu. Zwrócenie uwagi na konsekwencje, jakie niesie nietypowe postępowanie związane z zaburzeniami odżywiania. Uświadomienie tego problemu może być pierwszym i bardzo ważnym krokiem do podjęcia i wprowadzenia zmian w zachowaniu i postępowaniu. Świadomość negatywnych zachowań i ich skutków, przyczyni się do refleksji i monitorowania własnego postępowania potencjalnych użytkowników i analizowania zaburzeń odżywiania. Należy doprecyzować, że dokonywanie procesu wczesnego automonitoringu nie jest substytutem podejmowania terapii mającej na celu wyleczenie zaburzeń odżywiania. Proces notowania swoich działań w zakresie żywienia i aktywności fizycznej we własnym urządzeniu (np. smartfonie) może eliminować czynnik wstydu, ponieważ w czasie dokonywania rejestru swoich poczynań, użytkownik aplikacji nie ma styczności z drugim człowiekiem, nie zachodzi potrzeba budowania relacji lub pozyskiwania zaufania. Jednak wszystkie te działania mogą przyczynić się do wczesnego podjęcia skutecznego leczenia. Autorska aplikacja może pomóc w monitorowaniu nawyków żywieniowych. Jednak ważne jest zrozumienie, że autorska aplikacja nie jest terapią i nie może zastąpić profesjonalnej pomocy terapeutycznej. Jej rola polega na ułatwieniu osobom z zaburzeniami odżywiania śledzenia swoich działań, ale to terapeuci powinni zajmować się głównym procesem leczenia.

ROZDZIAŁ II
ANALIZA DOSTĘPNYCH APLIKACJI MOBILNYCH

W tej części pracy zostanie przedstawiony temat rynku aplikacji mobilnych i ich powiązania z zaburzeniami odżywiania, definicje aplikacji typu Mood Tracker oraz uzasadnienie potrzeby stworzenia autorskiej aplikacji. Celem przedstawienia powyższego tematu jest uświadomienie czytelnikowi deficytu na rynku aplikacji mobilnych w ujęciu funkcjonalności wczesnego ostrzegania przed zaburzeniami odżywiania, dla osób z zaburzeniami odżywiania, oraz korzyści wynikające z utworzenia własnej aplikacji mobilnej typu Mood Tracker.

2.1. Rynek aplikacji

Rynek programów przeznaczonych dla osób z zaburzeniami odżywiania jest coraz bardziej rozwinięty. Są dostępne aplikacje, które pomagają w monitorowaniu jedzenia i liczenia kalorii, które mogą pomóc osobom z zaburzeniami odżywiania w utrzymaniu zdrowych nawyków żywieniowych. Niektóre z nich oferują również wsparcie społeczne, takie jak przekierowanie do grup dyskusyjnych i możliwość komunikacji z terapeutami lub innymi specjalistami zdrowia. Wiele z tych mobilnych aplikacji jest bezpłatnych, ale niektóre oferują dodatkowe funkcje za opłatą. [46]

Aplikacje mobilne do liczenia kalorii, takie jak My Fitness Pal, są popularne wśród osób, które chcą monitorować swoje spożycie kalorii i dostosowywać swoją dietę do swoich celów fitness. Mogą one pomóc w utrzymaniu zdrowej wagi i poprawie ogólnego stanu zdrowia, są one dostępne na różne platformy, takie jak iOS i Android, i są łatwe w użyciu. Mają one zarówno pozytywne jak i negatywne strony, na podstawie badań z roku 2018 dotyczących wpływu aplikacji do liczenia kalorii My Fitness Pal na rozwój zaburzeń odżywiania odkryto, że znaczny odsetek uczestników korzystających z My Fitness Pal (73%) postrzegało aplikację, jako przyczyniającą się do ich zaburzeń odżywiania[47]. Istnieje związek pomiędzy problematycznym używaniem telefonu komórkowego, a objawami zaburzeń odżywiania[48].

[46] Źródło: Opracowanie własne na podstawie: Zasoby portalu Google Play, https://play.google.com/store/search?q=eating%20disorder&c=apps&hl=pl, dostęp: 05.02.2023
[47] C.A Levinson, L. Fewell, L.C. Brosof, *My Fitness Pal calorie tracker usage in the eating disorders*, „Eating Behaviors", nr 27, 2017, s. 14-16
[48] S. Li, G. Cui, Y. Yin, K. Tang, L. Chen, X. Liu, *Prospective Association Between Problematic Mobile Phone Use and Eating Disorder Symptoms and the Mediating Effect of Resilience in Chinese College Students: A 1-Year Longitudinal Study*, „Frontiers in Public Health", nr 10, 2022, s. 4

2.2. Aplikacje typu Mood Tracker

Aplikacje typu Mood Tracker są narzędziem do śledzenia nastroju. Pozwalają użytkownikom na codzienne rejestrowanie swoich emocji, w celu uzyskania lepszego rozeznania w swoich doświadczeniach emocjonalnych. Użytkownicy mogą wykorzystać tę aplikację do określenia wzorców nastroju, związanych z różnymi czynnikami, takimi jak dieta, ćwiczenia, relacje i wiele innych. Aplikacje te często zawierają również funkcje takie, jak przypomnienia, by rejestrować swój nastrój oraz analizę danych, aby pomóc użytkownikom zrozumieć swoje nastroje i lepiej je kontrolować[49].

Aplikacje typu Mood Tracker posiadają takie korzyści jak:

1. łatwy dostęp do informacji o nastroju: użytkownicy mogą łatwo rejestrować swoje nastroje i przeglądać historię swoich doświadczeń emocjonalnych;
2. zwiększenie świadomości nastroju: dzięki codziennemu rejestrowaniu nastroju, użytkownicy mogą lepiej zrozumieć, jak różne czynniki wpływają na ich emocje i jakie są ich naturalne wzorce nastroju;
3. pomoc w kontrolowaniu nastroju: aplikacja może pomóc użytkownikom w identyfikacji czynników, które mogą poprawić ich nastrój i zwiększyć ich zdolność do radzenia sobie z trudnymi emocjami;
4. analiza danych: aplikacja może zawierać narzędzia do analizy danych, które pomagają użytkownikom zrozumieć swoje nastroje i wzorce, co pomaga im w podejmowaniu lepszych decyzji dotyczących ich zdrowia i dobrego samopoczucia.

Tabela 13. Przegląd najpopularniejszych aplikacji typu *Mood Tracker* dla systemu Android

Nazwa	**Główne funkcjonalności**
Daylio - Dziennik, Pamiętnik	• Dokonywanie zapisu stanu danego nastroju na określony dzień (w formie 5 emoji) • Możliwość zarejestrowania predefiniowanych aktywności • Funkcja statystyk (rozszerzenie funkcjonalności poprzez dopłatę) • Personalizacja wizualna (rozszerzenie funkcjonalności poprzez

[49] S. M. Schueller, M. Neary, J. Lai, D. A. Epstein, *Understanding People's Use of and Perspectives on Mood Tracking Apps: An Interview Study (Preprint)*, „ JMIR Mental Health", 2021, s.17.

	dopłatę) • Ustawienie przypomnień • Wyznaczenie celu • Osiągnięcia • Zabezpieczenie dostępu w postaci pinu, odcisku palca • Płatna opcja kopii zapasowej
Moodpresss - Monitor nastrojów	• Dokonywanie zapisu stanu danego nastroju na określony dzień (w formie 11 emoji) • Możliwość zarejestrowania predefiniowanych lub niestandardowych aktywności • Możliwość eksportu pamiętnika do pliku PDF • Funkcja statystyk (rozszerzenie funkcjonalności na statystyki roczne poprzez dopłatę) • Personalizacja wizualna (rozszerzenie funkcjonalności poprzez dopłatę) • Ustawienie przypomnień • Wyznaczenie celu • Osiągnięcia • Zabezpieczenie dostępu w postaci pinu, odcisku palca • Opcja kopii zapasowej na dysku google
Moodflow: Mood Tracker	• Dokonywanie zapisu oceny danego dnia w skali od 1 do 5 • Możliwość zarejestrowania aktywności, stanów emocjonalnych, lokalizacji, przyjmowanych medykamentów, zapis interakcji społecznych. • Płatna funkcja pozwalająca wyznaczyć korelację między pogodą, a stanami emocjonalnymi • Funkcja porównania poprzednich okresów do bieżącego (tydzień, miesiąc, rok) • Płatna funkcja rejestracji geolokalizacji • Funkcja statystyk (rozszerzenie funkcjonalności poprzez dopłatę) • Zabezpieczenie dostępu w postaci pinu • Personalizacja wideo w tle

Mood Tracker: Self-Care Habits	• Dokonywanie zapisu stanu danego nastroju na określony dzień (w formie 5 emoji) • Możliwość zarejestrowania predefiniowanych lub niestandardowych aktywności • Możliwość doprecyzowania, co jest powodem danego nastroju na dzień • Płatna możliwość nagrania dźwięku, głosu jako formy rejestru • Wskazania co do dobrych nawyków • Rejestracja zwyczajów • Synchronizacja i tworzenie kopii zapasowych
Symptom & Mood Tracker	• Możliwość połączenia z Google Fit • Rejestracja objawów takich jak np. problemy z sercem, niski apetyt, refluks • Dokonywanie zapisu oceny danego dnia w skali od 1 do 5 • Eksport danych w formie elektronicznej • Kalendarz • Personalizacja wyglądu • Śledzenie lęków, bólu • Uwzględnienie symptomów takich chorób jak np. schorzeniami psychicznymi i fizycznymi, w tym lękiem, depresją, CFS (ME), SM (stwardnieniem rozsianym), fibromialgią, endometriozą, chorobą dwubiegunową, BPD, PTSD, migreny, bóle głowy, zawroty głowy, nowotwory (objawy chemioterapii), zapalenie stawów, choroba Crohna, cukrzyca, IBS i IBD.
DailyBean: Simplest Journa	• Dokonywanie zapisu stanu danego nastroju na określony dzień (w formie 5 fasolek) • Możliwość zarejestrowania predefiniowanych lub niestandardowych aktywności • Statystyki analizujące nastrój i aktywność tygodniowo/miesięcznie
eMoods Bipolar Mood Tracker	• Wysyłanie miesięcznego raportu PDF • Śledzenie zmian w lekach • Wysyłanie e-mailem raportu w formacie PDF pod koniec każdego miesiąca

	• Stworzona pod kątem choroby afektywnej dwubiegunowej, depresji i innych zaburzeń nastroju i lęku.
Tochi - Mood Tracker, Journal	• Dokonywanie zapisu oceny danego dnia w formie animowanych kulek • Aplikacja ma formę dziennika CBT do terapii poznawczo -behawioralnej • Statystyki dotyczące nastroju i zdrowia psychicznego • Możliwość zanotowania wyzwalaczy, lęków paniki
Wysa: Anxiety, therapy chatbot	• Chat bot AI w formie pingwina • Możliwość wyboru sposobu funkcjonowania programu (inwazyjna lub we własnym zakresie) • Płatna możliwość zabookowania 4 sesji z terapeutą • Dobór ćwiczeń w zależności od wybranych kryteriów • Rejestrowanie głosu w okresie lęku
Pixels - Mood & Mental Health	• Dokonywanie zapisu stanu danego nastroju na określony dzień (w formie 5 emoji) • Funkcja statystyk (ostatni tydzień, ostatnie 30 dni) • Ustawienie przypomnień • Zabezpieczenie dostępu w postaci pinu

Źródło: Opracowanie własne na podstawie: Zasobów portalu Google Play, https://play.google.com/store/search?q=mood+tracker&c=apps&hl=en_US&gl=US, dostęp: 05.02.2023

W tabeli (Patrz Tabela 13. Przegląd najpopularniejszych aplikacji Mood Tracker dla systemu Android), dokonano analizy najpopularniejszych aplikacji typu Mood Tracker występujących na platformie Google Play, żadne z zamieszczonych tam rozwiązań nie posiada opcji wykrywania zaburzeń odżywiania.

2.3. Uzasadnienie potrzeby stworzenia aplikacji

Wraz z rozwojem technologii na świecie, zmieniają się sposoby notowania pomysłów, rozwiązań. Notatniki, zeszyty z zapiskami swoich myśli, pomysłów, doświadczeń, są dzisiaj nieaktualną formą rejestrowania postępów czy zmian.

Głównymi wadami powyższej formy są:

- **Niedostępność:** zeszyty i notatniki są dostępne tylko w jednym miejscu, co uniemożliwia dostęp do nich, poza tym miejscem, co jest szczególnie problematyczne np. dla pacjentów, którzy muszą być pod opieką lekarza czy terapeuty;
- **Trudności w analizie danych:** ręczne notowanie danych jest czasochłonne i trudne do analizowania, co uniemożliwia interpretację danych i podejmowanie decyzji dotyczących leczenia;
- **Nieefektywność:** notowanie ręczne jest mniej efektywne niż elektroniczne, ponieważ pacjent, bądź osoba potencjalnie cierpiąca na zaburzenie odżywiania może łatwo pomylić się lub umyślnie wpisać coś co nie jest istotne w kontekście danej formy rejestrowania informacji. Brak określonych reguł co do formy notowania, może być przyczyną, że mimo starań dołożonych przez osobę potencjalnie chorą, dokonane zapisy mogą okazać się bezużyteczne do analizy. Istnieje także możliwość zapomnienia o notowaniu nastroju;

Należy pamiętać o tym, że wcześnie rozpoczęte leczenie zazwyczaj wróży sukces terapii zaburzeń odżywiania[50]. Skuteczna profilaktyka jest optymalnym sposobem hamowania rozwoju lub ograniczania skali zjawisk uznanych za niekorzystne i dolegliwe społecznie[51]. Takie choroby jak bulimia lub anoreksja są tematem tabu w społeczeństwie, przez co diagnoza pojawia się później bądź wcale, a w konsekwencji znacznie utrudnione jest wdrożenie stosownych kroków mających nieść pomoc[52].

Stworzenie autorskiej aplikacji mobilnej Mood Tracker z wczesnym wykrywaniem zaburzeń odżywiania, pozwoli na szerokie uświadomienie społeczeństwu problemu. Rozpowszechniona aplikacja może znacznie przyczynić się do wprowadzenia wczesnej profilaktyki zapobiegania zaburzeniom odżywiania. Ogólnodostępność aplikacji, popularność i łatwość notowania niekorzystnych procesów zachodzących w organizmie człowieka, pozwoli osobie potencjalnie mającej ten problem, do podjęcia stosownych działań i zastosowania skutecznych metod leczenia.

[50] J. Lock, D. Grange, *Twoje dziecko i zaburzenia odżywiania. Jak mu pomóc?*, Wydawnictwo Znak, Kraków 2006, s. 175.
[51] D. Mroczkowska, B. Ziółkowska, A. Cwojdzińska, *Zaburzenia odżywiania* …, s. 103.
[52] K. Gaber, B. Kuk, *Kiedy…*, s. 42

ROZDZIAŁ III

AUTORSKA APLIKACJA *MOOD TRACKER*

W tej części pracy omówiono zagadnienia dotyczące użytych technologii, funkcjonalności, przedstawienia i omówienia najważniejszych fragmentów kodu źródłowego, instrukcje obsługi oraz prezentacje wyników analizy strategicznej SWOT. Celem tego rozdziału jest przedstawienie aplikacji, jej głównych funkcji oraz analiza potencjalnych zalet i ryzyk związanych z proponowanym rozwiązaniem.

3.1. Użyte technologie

Aplikacja autorska Mood Tracker została rozwinięta w środowisku Android Studio Flamingo 2022, minimalna obsługiwana wersja systemu Android to 8.0 lub nowsza.

Powyższa aplikacja mobilna opiera się na platformie Firebase, dostarczając kompleksowe narzędzia do budowy aplikacji mobilnych. W tym kontekście, Firebase Authentication został użyty do bezpiecznego uwierzytelniania użytkowników, co pozwala na kontrolę dostępu do aplikacji oraz zabezpieczenie prywatnych danych użytkowników. Dodatkowo, wykorzystano rozwiązania Firestore, które jest zaawansowaną *bazą danych* typu *NoSQL*. To rozwiązanie w chmurze umożliwia przechowywanie danych w czasie rzeczywistym oraz synchronizację między różnymi urządzeniami mobilnymi. Dane w *Cloud Firestore* są przechowywane w postaci *struktur dokumentów*, co pozwala na bardziej elastyczną strukturę przechowywania danych.

3.2. Wymagania funkcjonalne i ich realizacja

Tabela 14. Wymagania funkcjonalne autorskiej aplikacji Moodtracker

Autentykacja i autoryzacja użytkownika
Sczytywanie nastrojów
Zapis nastrojów
Analityka posiadanych rekordów
Powiadomienia odnośnie do potencjalnych zaburzeń
Praca w trybie offline
Obsługa błędów
Tłumaczenie zasobów

Źródło: Opracowanie własne

W tabeli (patrz Tabela 14. Wymagania funkcjonalne autorskiej aplikacji Moodtracker), wymieniono najważniejsze wymagania, jakie autorska aplikacja Mood Tracker spełnia.

Aby umożliwić użytkownikowi *autentykację*, skorzystano z usługi *mechanizmu uwierzytelniania Firebase*, który umożliwia wykorzystanie *konta Google* do procesu logowania. Decyzja o wyborze powyższego rozwiązania jest podyktowana faktem, że Google jest głównym dystrybutorem aplikacji dla urządzeń z systemem Android, poprzez *Sklep Google Play*.[53] Dzięki temu użytkownicy posiadający konta Google mogą logować się do aplikacji, co przyczynia się do płynnej *interakcji z nią* oraz do wykorzystania istniejących danych logowania.

Dla zapewnienia połączenia pomiędzy aplikacją a platformą Firebase, dodano plik konfiguracyjny, który zawiera *klucz API*, służący do *autoryzacji* aplikacji do korzystania z usług Firebase. W celu zwiększenia bezpieczeństwa procesów interakcji z bazą danych, do konfiguracji bazy dodano indywidualny *klucz SHA256* aplikacji (*Secure Hash Algorithm Certificate fingerprint*), który pełni rolę *unikalnego identyfikatora* pozwalającego *uwierzytelnić i autoryzować* aplikację do komunikacji z usługami Firebase.

W celu umożliwienia użytkownikom *uwierzytelniania* i dostępu do aplikacji, wykorzystano zestaw narzędzi dostarczonych przez platformę Firebase. Dzięki temu podejściu, proces logowania jest bardziej usprawniony, a użytkownicy mogą korzystać z własnych kont Google, co przyczynia się do poprawy komfortu korzystania z aplikacji.

Tabela 15. Listing kodu odpowiedzialnego za tworzenie opcji konfiguracyjnych logowania

```
val googleSignInOptions =
GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
    .requestIdToken(getString(R.string.default_web_client_id))
    .requestEmail()
    .build()
```

Źródło: Opracowanie własne, metoda OnCreate w definicji klasy MainActivity.

Opisany w tabeli (patrz Tabela 15. Listing kodu odpowiedzialnego za tworzenie opcji konfiguracyjnych logowania), fragment jest kluczowy, ponieważ jest wykorzystywany do *konfiguracji procesu logowania* w aplikacji. Tworzy on domyślne ustawienia logowania, które obejmują *żądanie uzyskania tokena identyfikacyjnego (ID token)* oraz *dostępu do adresu e-mail użytkownika* podczas logowania. Dzięki temu, aplikacja może skutecznie zarządzać procesem uwierzytelniania użytkowników i zbierać niezbędne informacje podczas logowania, co jest istotne dla funkcjonowania mechanizmu logowania w aplikacji.

[53] Opracowanie na podstawie: Zasoby portalu Android, https://www.android.com/intl/pl_pl/everyone/facts/, dostęp: 19.08.2023

Tabela 16. Listing kodu odpowiedzialnego za tworzenie klienta Google API

```
googleApiClient = GoogleApiClient.Builder(this)
    .addApi(Auth.GOOGLE_SIGN_IN_API, googleSignInOptions).build()
```

Źródło: Opracowanie własne, metoda OnCreate w definicji klasy MainActivity

Przedstawiony w tabeli (patrz Tabela 16. Listing kodu odpowiedzialnego za tworzenie klienta Google API), kod jest istotny, ponieważ umożliwia aplikacji obsługę procesu logowania z wykorzystaniem konta Google. Pozwala to użytkownikom na logowanie oraz zapewnia dostęp do usług Firebase. Tworzenie klienta *Google API* w *aktywności głównej* (*MainActivity*) jest kluczowe, ponieważ ona jedyną gdzie, wykonywane są operacje związane z powyższymi usługami. Klient ten jest konfigurowany z wcześniej zdefiniowanymi opcjami logowania (patrz Tabela 15. Listing kodu odpowiedzialnego za tworzenie opcji konfiguracyjnych logowania), co pozwala na uwierzytelnianie użytkowników. Gotowy *obiekt klienta* jest następnie używany do realizacji operacji logowania, umożliwiając użytkownikom dostęp do aplikacji przy użyciu swojego konta Google.

Aplikacja posiada dwa menu nawigacji, zlokalizowane odpowiednio w *dolnym pasku nawigacyjnym* (*Bottom Navigation Bar*), służące do zmiany między 5 *fragmentami* oraz w *pasku akcji* (*Support Action Bar*) w górnej części aktywności, służące do obsługi logowania.

Tabela 17. Listing kodu odpowiedzialnego za obsługę nasłuchiwania wyboru menu logowania

```
navDrawer.setNavigationItemSelectedListener { menuItem ->
   when (menuItem.itemId) {
      R.id.navigation_Login -> {
         val signInIntent = Auth.GoogleSignInApi.getSignInIntent(googleApiClient)
         startActivityForResult(signInIntent, RC_SIGN_IN)
         val currentUser = FirebaseAuth.getInstance().currentUser
         updateUI(currentUser)
         return@setNavigationItemSelectedListener true
      }
R.id.navigation_Logout -> {
                val firebaseAuth = FirebaseAuth.getInstance()
                firebaseAuth.signOut()
                val currentUser = FirebaseAuth.getInstance().currentUser
                updateUI(currentUser)
                val intent = Intent(this, MainActivity::class.java).apply { flags=
Intent.FLAG_ACTIVITY_NEW_TASK or Intent.FLAG_ACTIVITY_CLEAR_TASK}
                startActivity(intent)
                finish()
                return@setNavigationItemSelectedListener true
             }
      R.id.navigation_About -> {/….
```

```
        }
    } false
```
Źródło: Opracowanie własne, metoda OnCreate w definicji klasy MainActivity

W tabeli powyżej (patrz Tabela 17. Listing kodu odpowiedzialnego za obsługę nasłuchiwania wyboru menu logowania), fragment kodu odpowiada za tworzenie predefiniowanej *intencji* z parametrem wejściowym poprzednio utworzonym *klientem* (patrz Tabela 16. Listing kodu odpowiedzialnego za tworzenie klienta Google API).

Tabela 18. Listing kodu odpowiedzialnego za obsługę odczytywania rezultatu intencji logowania

```
override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {
    super.onActivityResult(requestCode, resultCode, data)
    if (requestCode == RC_SIGN_IN) {
        val result = Auth.GoogleSignInApi.getSignInResultFromIntent(data!!)
        if (result!!.isSuccess) {
            val account = result.signInAccount
            firebaseAuthWithGoogle(account)
        } else {
            toastShow(getString(R.string.sign_in_failed))
        }}}
```
Źródło: Opracowanie własne, metoda OnCreate w definicji klasy MainActivity

W przedstawionej tabeli (patrz Tabela 18. Listing kodu odpowiedzialnego za obsługę odczytywania rezultatu intencji logowania), opisano kluczową procedurę *obsługi wyniku procesu logowania*. Ten fragment kodu jest fundamentalny, ponieważ odpowiada za *uwierzytelnienie użytkownika* w Firebase po pomyślnym zalogowaniu się przez konto Google. To kluczowy krok, który umożliwia aplikacji dostęp do zasobów i funkcji powiązanych z konkretnym użytkownikiem, co może znacząco wpłynąć na działanie i funkcjonalność aplikacji.

Tabela 19. Listing kodu odpowiedzialnego za obsługę logowanie przy użyciu pobranego tokena

```
private fun firebaseAuthWithGoogle(account: GoogleSignInAccount?) {
    val idToken = account?.idToken
    if (idToken != null) {
        val credential = GoogleAuthProvider.getCredential(idToken, null)
        FirebaseAuth.getInstance().signInWithCredential(credential)
            .addOnCompleteListener(this) { task ->
                if (task.isSuccessful) {
                    val user = FirebaseAuth.getInstance().currentUser
                    val email = user?.email
                    val toastString = buildString {append(getString(R.string.signed_in_as))
    append(email)
```

```
}
                updateUI(user)
                toastShow(toastString)
            } else {
                toastShow(getString(R.string.sign_in_failed))
            }
        }
    } else {
        toastShow(getString(R.string.unable_to_retrieve_idtoken))
    }
}
```

Źródło: Opracowanie własne, metoda prywatna firebaseAuthWithGoogle w definicji klasy MainActivity

W powyższej tabeli (patrz Tabela 19. Listing kodu odpowiedzialnego za obsługę logowanie przy użyciu pobranego tokena), przedstawiona została metoda, która odgrywa istotną rolę w procesie uwierzytelniania użytkowników, ponieważ odpowiada za przetwarzanie pobranego „*idToken*", dzięki temu użytkownicy mogą być uwierzytelniani w opisanej usłudze, co umożliwia im dostęp do *zasobów i funkcji aplikacji*.

Logowanie z użyciem poświadczenia jest *zadaniem* w oddzielnym wątku. Kod używa nasłuchiwacza, który w przypadku zakończenie tego zadania sukcesem, pobiera informacje o użytkowniku i dostosowuje interfejs graficzny. W przypadku błędu, użytkownik otrzymuje komunikat o niepowodzeniu.

Tabela 20. Listing kodu odpowiedzialnego za zasady bazy danych Cloud Firestore

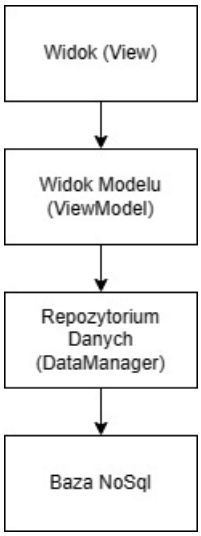
```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /Moods/{userId}/MoodEntries/{entryId} {
      allow read, write: if request.auth != null && request.auth.uid == userId;
    }
    match /Profiles/{userId} {
      allow read, write: if request.auth != null && request.auth.uid == userId;
    }
    match /Notifications/{userId}/NotificationEntries/{entryId} {
      allow read, write: if request.auth != null && request.auth.uid == userId;
    }
    match /Symptoms/{document=**} {
      allow read: if request.auth != null;
      allow write: if false; // Odmawia dostępu do zapisu
    }
    match /{document=**} {allow read, write: if false;
```

Źródło: Opracowanie własne, zasady bazy danych Cloud Firebase dla aplikacji Mood Tracker

Poprzedzająca tabela (patrz Tabela 20. Listing kodu odpowiedzialnego za zasady bazy danych Cloud Firestore), przedstawia fragment kodu opisujący zasady dostępu (*data rules*) do użytej bazy danych w autorskiej aplikacji Mood Tracker.Jest on niezbędny,

ponieważ pozwala on uwierzytelnionym użytkownikom na odczyt i zapis własnych wpisów nastrojów, profilu i powiadomień, a jednocześnie ograniczając dostęp do innych danych i dokumentów w bazie.

Rysunek 2. Struktura danych aplikacji Mood Tracker



Źródło: Opracowanie własne

Na ilustracji (patrz Rysunek 2 Struktura danych aplikacji Mood Tracker) przedstawiono sposób organizacji kodu zastosowany w aplikacji w *architekturze „Modelu – Widoku - Widoku Modelu”*.

Tabela 21. Listing kodu odpowiedzialnego za model używany do przechowywania nastroju

```
data class Mood(
    val description: String,
    val createDate: Date,
    val timeZone: ZoneId,
    val rating: Int,
    val symptomList: List<Symptom>?,
    val key: String? = null
) : Parcelable {}
```

Źródło: Opracowanie własne, definicja klasy Mood.

W tabeli (patrz Tabela 21. Listing kodu odpowiedzialnego za model używany do przechowywania nastroju), opisano *model* reprezentujący nastroje użytkownika, jest ważny ponieważ służy do przekazywania pobranych wartości w logice aplikacji.

Zawiera on właściwości:

- „*description: String*” - opis nastroju;
- „*createDate: Date*” - data utworzenia nastroju;
- „*timeZone: ZoneId*” - strefa czasowa, w jakiej utworzono nastrój;

- „*rating: Int*” - ocena nastroju w skali od 1 do 5;
- „*symptomList: List<Symptom>?*” - opcjonalna lista objawów związanych z nastrojem;
- „*key: String?*” - opcjonalny *identyfikator klucza* w bazie wykorzystywany do operacji aktualizacji rekordów. Powyższa klasa *implementuje interfejs Parcelable*, umożliwiający *serializację* i *deserializację*, wykorzystywaną do celów przekazywania obiektów jako *argument* za pomocą *akcji* między *fragmentami* w autorskiej aplikacji.

Tabela 22. Listing kodu odpowiedzialny za sczytanie nastrojów z bazy

```
fun readMoods(callback: MoodsCallback) {
    val currentUser: FirebaseUser? = FirebaseAuth.getInstance().currentUser
    if (currentUser != null) {
        val moodsRef =
firestore.collection("Moods").document(currentUser.uid).collection("MoodEntries")
        moodsRef.get().addOnSuccessListener { querySnapshot ->
                val moods = mutableListOf<Mood>()
                for (document in querySnapshot) {
                    val mood = mapDocumentToMood(document)
                    moods.add(mood)
                }
                callback.onMoodsLoaded(moods)
            }
            .addOnFailureListener { _ ->
                callback.onError(ErrorCode.EMR)
            }
    } else {
        callback.onError(ErrorCode.EMA)
    }
}
```

Źródło: Opracowanie własne, metoda readMoods z klasy Datamanger

W powyższej tabeli (patrz Tabela 22. Listing kodu odpowiedzialny za sczytanie nastrojów z bazy), przedstawiono metodę odpowiedzialną za odczyt nastrojów z bazy danych. Jako argumenty przyjmuje ona *interfejs* obsługujący dwa zdarzenia (sukcesu odczytu i jego braku). Pierwszym krokiem jest sprawdzenie, czy użytkownik został uwierzytelniony. Następnie tworzona jest *referencja do kolekcji* przypisanej użytkownikowi. Na *referencji* używana jest metoda „*get”*, która może osiągnąć rezultaty w postaci:

- *nasłuchiwacz sukcesu* odnotuje zdarzenie pomyślnego odczytu, następuje *iteracja* przez wyniki i wywołanie funkcji *interfejsu* odpowiedzialną za pomyślny odczyt z argumentem nastrojów.
- *nasłuchiwacz niepowodzenia* zanotuje negatywny wynik operacji, wywołanie funkcji interfejsu odpowiedzialną za błąd odczytu z argumentem odpowiadającym

odpowiedniemu kodowi błędu. Ujednolicony kod błędów opisany w tabeli (patrz Tabela 49. Listing kodu klasy enumerowanej ErrorCode)

Tabela 23. Fragment kodu odpowiedzialny za pomyślne sczytanie nastrojów z bazy

```
.addOnSuccessListener { querySnapshot ->
                val moods = mutableListOf<Mood>()
                for (document in querySnapshot) {
                    val mood = mapDocumentToMood(document)
                    moods.add(mood)
                }
                callback.onMoodsLoaded(moods)
            }
```

Źródło: Opracowanie własne, fragment metody readMoods w definicji klasy DataManager

W przedstawionej tabeli (patrz Tabela 23. Fragment kodu odpowiedzialny za pomyślne sczytanie nastrojów z bazy), znajduje się fragment kodu odpowiedzialny za dodanie *nasłuchiwacza sukcesu* na operacji „*get*". *Wynikiem zapytania* jest *migawka zapytania* użyta w *funkcji lambda*, w niej następuje iteracja przez każdy *dokument* z *wyniku kwerendy*, *mapowanie* go do *modelu* i dodanie do wcześniej utworzonej pustej *kolekcji*.

Tabela 24. Listing kodu odpowiedzialnego za mapowanie dokumentu do modelu

```
private fun mapDocumentToMood(document: DocumentSnapshot): Mood {
    val description = document.getString("description") ?: ""
    val createDate = document.getDate("createDate") ?: Date()
    val timeZoneMap = document.get("timeZone") as? Map<String, Any>
    val timeZoneId = timeZoneMap?.get("id") as? String ?: "UTC"
    val timeZone = ZoneId.of(timeZoneId)
    val rating = document.getLong("rating")?.toInt() ?: 0
    val symptomsSnapshot = document.get("list") as? List<Map<String, Any>>?
    val symptomsList = symptomsSnapshot?.mapNotNull { symptomMap ->
        val text = symptomMap["text"] as? String
        val id = symptomMap["id"] as? String
        text?.let { Symptom(it, id ?: "") }
    }
    val key = document.id
    return Mood(description, createDate, timeZone, rating, symptomsList, key)
}
```

Źródło: Opracowanie własne, definicja funkcji mapDocumentToMood w klasie DataManager

W tabeli (patrz Tabela 24. Listing kodu odpowiedzialnego za mapowanie dokumentu do modelu), przedstawiono funkcję odpowiedzialną za mapowanie dokumentu do modelu. Metoda ta jest kluczowa ponieważ jest używana przy każdej *iteracji dokumentów* odpowiadającym nastrojom.

Tabela 25. Listing kodu interfejs MoodCallback

```
interface MoodsCallback {
    fun onMoodsLoaded(moods: List<Mood>)
    fun onError(error: ErrorCode) }
```

Źródło: Opracowanie własne, definicja interfejsu MoodCallback

W powyższej tabeli (patrz Tabela 25. Listing kodu interfejs MoodCallback) znajduję się kod *Interfejsu*, wykorzystywany do przekazywania wyników pobierania nastrojów oraz informacji o błędach. W powyższej definicji zdefiniowane są dwie funkcje:

- Metoda obsługującą pomyślne pobranie;
- Metoda obsługującą błąd pobrania;

Powyższy kod wykorzystywany jest ważny, ponieważ jest wykorzystywany przy pobraniu nastrojów przy użyciu *klasy anonimowej*.

Tabela 26. Listing kodu odpowiedzialny za sczytanie nastrojów z bazy

```
    fun readMoodFromDay(date: LocalDate, timeZone: ZoneId, callback: (List<Mood>) -> Unit) {
        val currentUser: FirebaseUser? = FirebaseAuth.getInstance().currentUser
        if (currentUser != null) { val moodsRef =
            firestore.collection("Moods").document(currentUser.uid).collection("MoodEntries")
            val startOfDay = date.atStartOfDay(timeZone).toInstant().toEpochMilli()
            val endOfDay = date.atTime(23, 59, 59,
999_999_999).atZone(timeZone).toInstant().toEpochMilli()
            moodsRef
                .whereGreaterThanOrEqualTo("createDate", Timestamp(startOfDay / 1000, 0))
                .whereLessThanOrEqualTo("createDate", Timestamp(endOfDay / 1000,
999_999_999)).get()
                .addOnSuccessListener { querySnapshot ->
                    val moods = mutableListOf<Mood>()
                    for (document in querySnapshot) {
                        val mood = mapDocumentToMood(document)
                        moods.add(mood)
                    }
                    callback(moods)
                }
                .addOnFailureListener {
                    callback(emptyList())
                }
        }
    }
```

Źródło: Opracowanie własne, definicja funkcji mapDocumentToMood w klasie DataManager

W powyższej tabeli (patrz Tabela 26. Listing kodu odpowiedzialny za sczytanie nastrojów z bazy), przedstawiono metodę odpowiedzialna za pobranie rekordów z danego dnia, jako *argumenty* przyjmuje ona:

- Datę z której mają zostać pobrane nastroje;
- *Identyfikator strefy czasowej*;

• *Funkcje zwrotna*, przetwarzająca listę nastrojów.

Pierwszym etapem jest sprawdzenie, czy użytkownik został uwierzytelniony. Następnie tworzona jest referencja do kolekcji przypisanej użytkownikowi. Kolejno ma miejsce obliczanie wartości czasu „chwili" rozpoczęcia dnia z argumentu i jego zakończenia, uwzględniając strefę czasową. Wartości te obejmują cały wybrany dzień od północy (*00:00:00.000*) do prawie północy (*23:59:59.999*). Na powyższej referencji definiowane jest zapytanie pobierające rekordy z tego dnia. Na zapytaniu używana jest metoda „*get*", która może osiągnąć wyniki w postaci:

• *nasłuchiwacz sukcesu* odnotuje zdarzenie pomyślnego odczytu, następuje wywołanie *funkcji zwrotnej* z pobranymi nastrojami jako argument.

• *nasłuchiwacz niepowodzenia* zanotuje negatywny wynik operacji, poprzedzone jest to wywołaniem zwrotnym z parametrem listą pustą.

Tabela 27. Listing kodu funkcji wczytywania nastrojów dla konkretnej daty

```
   fun readSelectedDateMood(selectedYear: Int, selectedMonth: Int, selectedDayOfMonth:
Int, callback: (List<Mood>) -> Unit
   ) {
      val selectedDate = LocalDate.of(selectedYear, selectedMonth, selectedDayOfMonth)
      val timeZone = ZoneId.systemDefault()
      dataManager.readMoodFromDay(selectedDate, timeZone) { moods ->
         val filteredMoods = moods.sortedByDescending { it.createDate }
         callback(filteredMoods)
      }}
```

Źródło: Opracowanie własne, funkcja readSelectedDateMood w MoodViewModel

W tabeli wyżej umieszczonej (patrz Tabela 27. Listing kodu funkcji wczytywania nastrojów dla konkretnej daty), znajduje się definicja funkcji odczyt nastrojów dla wybranej daty, pobranej z *widoku*.

Funkcja przyjmuje trzy argumenty:

- Wybrany rok;
- Wybrany miesiąc;
- Wybrany dzień miesiąca;
- *Funkcja zwrotna*, przetwarzająca listę nastrojów.

Wewnątrz funkcji, tworzona jest data na podstawie powyższych *argumentów*, a następnie odczytywane są nastroje utworzone tego dnia przy użyciu metody odpowiedzialnej za odczyt nastrojów z danej daty. Następnie nastroje są sortowane według czasu utworzenia i przekazywane jako argument w użytym wywołaniu zwrotnym.

Tabela 28. Listing kodu funkcji tworzenia obiektu klasy Mood

```
fun createMood(
        moodCreate: Mood,
        successCallback: () -> Unit,
        errorCallback: (ErrorCode) -> Unit
    ) {val dataManager = DataManager()
        dataManager.getLastMoodAddedTime { lastAddedTime ->
            if (lastAddedTime != null) {val currentTime = Date()
                val timeDifference = currentTime.time - lastAddedTime.time
                val minutesDifference = TimeUnit.MILLISECONDS.toMinutes(timeDifference)
                if (minutesDifference >= 15) {
                    dataManager.createMood(
                        moodCreate,
                        onSuccess = {
                            moodAnalisis.analyzeMoods()
                            successCallback()},
                        onError = { error ->
                            errorCallback(error)
                        }
                    )
                } else { errorCallback(ErrorCode.EMC15)
                }
            } else {
                dataManager.createMood(
                    moodCreate,
                    onSuccess = {
                        successCallback()
                    },
                    onError = { error ->
                        errorCallback(error)
                    })
            }
        }}
```

Źródło: Opracowanie własne, fragment metody create mood w AddMoodViewModel

W tabeli (patrz Tabela 28. Listing kodu funkcji tworzenia obiektu klasy Mood), opisano funkcje odpowiedzialną za dodanie rekordu w widoku modelu, jako argumenty przyjmuje ona:

- *Obiekt* reprezentujący nastrój pobrany z *widoku*;
- *Funkcje zwrotną*, która zostanie wywołana w przypadku pomyślnego dodania;
- *Funkcje zwrotną*, która zostanie wywołana w przypadku wystąpienia błędu;

Na utworzonym obiekcie klasy odpowiedzialnej za *repozytorium danych*, wywoływana jest metoda odpowiedzialna za uzyskanie informacji o czasie ostatniego dodanego nastroju przez użytkownika. W przypadku gdy pobrany czas nie jest *null*, obliczana jest różnica czasu między bieżącym czasem a czasem ostatniego nastroju w minutach. Następnie sprawdzane jest, czy różnica czasu wynosi co najmniej 15 minut. Jeśli obliczony czas jest starszy niż 15 minut, bądź jest równy *null (informacja o braku poprzednio dodanych rekordów),*

to wywoływana jest metoda odpowiedzialna za dodanie rekordu do bazy. Jeśli operacja ta zakończy się sukcesem, wywoływana jest funkcja odpowiedzialna za analizę rekordów w bazie, a następnie *wywołanie zwrotne* przypisane sukcesowi. Przeciwnym wypadku ma miejsce użycie *funkcji zwrotnej*, przypisane wystąpieniu błędu z przekazaniem informacji o błędzie.

Tabela 29. Listing kodu funkcji odpowiadzialnej za sprawdzenie ostatniego dodanego rekordu

```
    fun getLastMoodAddedTime(callback: (Date?) -> Unit) {
        val currentUser: FirebaseUser? = FirebaseAuth.getInstance().currentUser
        if (currentUser != null) {
            val moodsRef =
firestore.collection("Moods").document(currentUser.uid).collection("MoodEntries")
            val query = moodsRef.orderBy("createDate", Query.Direction.DESCENDING).limit(1)
            query.get()
                .addOnSuccessListener { querySnapshot ->
                    if (querySnapshot.documents.firstOrNull() != null){
                        val lastMood = mapDocumentToMood(querySnapshot.documents.firstOrNull()!!)
                        callback(lastMood.createDate)
                    }
                    else {
                        callback(null)
                    }
                }
                .addOnFailureListener {
                    callback(null)
                }
        } else {callback(null)}}
```

Źródło: Opracowanie własne, funkcja getLastMoodAddedTime klasy DataManager

W wyżej opisanej tabeli (patrz Tabela 29. Listing kodu funkcji odpowiedzialnej za sprawdzenie ostatniego dodanego rekordu), przedstawiono metodę odpowiedzialna za sprawdzenie czasu ostatnio dodanego rekordu, przyjmuje ona parametr *funkcji zwrotnej* z *argumentem* daty dopuszczającym wartość *null*.

Pierwszym etapem jest sprawdzenie, czy użytkownik został uwierzytelniony. Następnie tworzona jest *referencja do kolekcji* przypisanej użytkownikowi. Na powyższym definiowane jest zapytanie pobierające najmłodszą datę z atrybutu daty utworzenia w dokumentach.

Na zapytaniu używana jest metoda *„get*”, która może osiągnąć wyniki w postaci:

- *nasłuchiwacz sukcesu* odnotuje zdarzenie pomyślnego odczytu, następuje wywołanie *funkcji zwrotnej* z pobraną datą jako argument.
- *nasłuchiwacz niepowodzenia* zanotuje negatywny wynik operacji, poprzedzone jest to *wywołaniem zwrotnym* z parametrem *null*.

Jeżeli użytkownik nie jest zalogowany, wywoływana jest *funkcja zwrotna* z argumentem *null*.

Tabela 30. Listing kodu funkcji odpowiedzialnej dodanie rekordu do bazy danych

```
    fun createMood(moodCreate: Mood , onSuccess: () -> Unit , onError: (ErrorCode) -> Unit) {
        val currentUser: FirebaseUser? = FirebaseAuth.getInstance().currentUser
        if (currentUser != null) {
            val moodsRef =
Firestore.collection("Moods").document(currentUser.uid).collection("MoodEntries")
            moodsRef.add(moodCreate)
                .addOnSuccessListener {
                    onSuccess()
                }
                .addOnFailureListener { _ ->
                    onError(ErrorCode.EMC)
                }
        } else {
            onError(ErrorCode.EMA)
        }
    }
```

Źródło: Opracowanie własne

W powyższej tabeli (patrz Tabela 30. Listing kodu funkcji odpowiedzialnej dodanie rekordu do bazy danych), opisano metodę odpowiedzialna za dodawanie nowego rekordu reprezentującego nastrój, przyjmuje ona argumenty:

- *Obiekt* reprezentujący nastrój, który ma zostać dodany do bazy danych;
- *Funkcja zwrotna*, która zostanie *wywołana* w przypadku pomyślnego dodania nastroju do bazy danych;
- *Funkcja zwrotna*, która zostanie *wywołana* w przypadku wystąpienia błędu podczas próby dodania nastroju;

Pierwszym krokiem jest sprawdzenie, czy użytkownik został uwierzytelniony. Następnie tworzona jest *referencja* do *kolekcji*, będąca przypisana użytkownikowi. Poprzez użycie metody odpowiedzialnej za dodanie rekordu, wywołanej na powyższej *referencji,* dodawany jest do niej nowy *dokument*. Powyższa operacja może osiągnąć rezultaty w postaci:

- *nasłuchiwacz sukcesu* odnotuje zdarzenie pomyślnego dodania do bazy, wywołuje *funkcje zwrotną*, przypisaną pomyślnemu rezultatowi.
- *nasłuchiwacz niepowodzenia* zanotuje negatywny wynik operacji, następuje *wywołanie zwrotne,* odpowiadające niepomyślnemu rezultatowi z kodem błędu, wskazującym na błąd podczas próby dodania nastroju.

Jeżeli użytkownik nie jest zalogowany, wywoływana jest funkcja zwrotna, obsługująca to zdarzenie.

Tabela 31. Listing kodu funkcji odpowiedzialnej za aktualizacje rekordu w widoku modelu

```
    fun updateMood(moodUpdate: Mood, successCallback: () -> Unit, errorCallback: (ErrorCode) ->
Unit) {
        dataManager.updateMood(moodUpdate,
            onSuccess = {
                successCallback()
            },
            onError = { error ->
                errorCallback(error)
            }
        )
    }
```

Źródło: Opracowanie własne, funkcja updateMood w widoku modelu AddMoodViewModel

W tabeli wyżej (Tabela 31. Listing kodu funkcji odpowiedzialnej za aktualizacje rekordu w widoku modelu), przedstawiono funkcje w widoku modelu, odpowiedzialną za aktualizacje rekordu reprezentującego nastrój. Ważna jest, ponieważ umożliwia komunikację między aplikacją a bazą danych, a także zapewnia elastyczność w *obsłudze błędów*.

Tabela 32. Listing kodu funkcji obsługującej edycje rekordu w bazie

```
    fun updateMood(moodUpdate: Mood, onSuccess: () -> Unit, onError: (ErrorCode) -> Unit) {
        val currentUser: FirebaseUser? = FirebaseAuth.getInstance().currentUser
        if (currentUser != null && moodUpdate.key != null) {
            val moodsRef =
firestore.collection("Moods").document(currentUser.uid).collection("MoodEntries")
            moodsRef.document(moodUpdate.key).set(moodUpdate)
                .addOnSuccessListener {
                    onSuccess()
                }
                .addOnFailureListener { _ ->
                    onError(ErrorCode.EMU)
                }
        } else {
            onError(ErrorCode.EMA)
        }
    }
```

Źródło: Opracowanie własne

Funkcja w tabeli powyżej (patrz Tabela 32. Listing kodu funkcji obsługującej edycje rekordu w bazie), odpowiada za aktualizację istniejącego rekordu nastroju w bazie danych. Jako argumenty powyższej metody przyjmuje się:

- „*moodUpdate*”: *Obiekt* typu *Mood*, który reprezentuje zaktualizowany nastrój.

- „*onSuccess*”: *Funkcja* zwrotna, która zostanie użyta w przypadku pomyślnego zaktualizowania nastroju w bazie danych.
- „*onError*”: *Wywołanie zwrotne*, która zostanie użyta w przypadku wystąpienia błędu podczas próby aktualizacji nastroju.

Metoda ta najpierw sprawdza, czy istnieje zalogowany użytkownik, wykonuje to poprzez pobranie *obiektu „currentUser”* z wykorzystaniem „*FirebaseAuth.getInstance().currentUser”*. Jeśli użytkownik jest zalogowany oraz jednocześnie istnieje *klucz* w obiekcie *„moodUpdate*”, funkcja przechodzi do procesu aktualizacji. Tworzona jest *referencja do kolekcji* nastrojów w bazie danych, przypisana dla aktualnie zalogowanego użytkownika. Za pomocą metody „*document(moodUpdate.key)*” wybierany jest konkretny dokument o *identyfikatorze klucza*. Wywołanie „*set*”, aktualizuje zawartość wybranego *dokumentu* nastroju na podstawie danych z *obiektu* „*moodUpdate*”.

Jeśli aktualizacja *dokumentu* zakończy się pomyślnie, *wywoływana* jest *funkcja zwrotna* „*onSuccess*”, sygnalizując sukces operacji. W przypadku wystąpienia błędu podczas aktualizacji *dokumentu* lub użytkownik nie jest zalogowany lub brakuje *identyfikatora* w *obiekcie,* wywoływana jest *funkcja zwrotna* odpowiadająca za obsługę błędu

Tabela 33. Listing kodu funkcji obsługującej edycje rekordu

```
    fun deleteMood( selectedMood: Mood, successCallback: () -> Unit, errorCallback: (ErrorCode) -
> Unit){
        val currentUser = FirebaseAuth.getInstance().currentUser
            val dataManager = DataManager()
            if (!selectedMood.key.isNullOrEmpty()){
                dataManager.deleteMood(
                    selectedMood.key,
                    onSuccess = {
                        successCallback()
                    },
                    onError = { error ->
                        errorCallback(error)
                    }
                )
        }
    }
```

Źródło: Opracowanie własne, metoda deleteMood w widoku modelu AddMoodViewModel

Funkcja z *widoku modelu* znajdująca się w tabeli (patrz Tabela 33. Listing kodu funkcji obsługującej usuwanie rekordu), sprawdza czy *unikalny klucz obiektu* przypisany automatycznie przez bazę nie jest pusty i przekazuje go do *repozytorium danych*.

Tabela 33. Listing kodu funkcji obsługującej usuwanie rekordu w bazie

```
fun deleteMood(id: String , onSuccess: () -> Unit , onError: (ErrorCode) -> Unit){
    val currentUser: FirebaseUser? = FirebaseAuth.getInstance().currentUser
    if (currentUser != null) {
        val moodsRef =
firestore.collection("Moods").document(currentUser.uid).collection("MoodEntries").document(id)
        moodsRef.delete()
            .addOnSuccessListener {
                onSuccess()
            }.addOnFailureListener { _ ->
                onError(ErrorCode.EMD)}
    } else {onError(ErrorCode.EMA) }
```

Źródło: Opracowanie własne

Funkcja *z tabeli* (patrz Tabela 33. Listing kodu funkcji obsługującej usuwanie rekordu w bazie), odpowiada za kluczową operacje usunięcia pojedynczego nastroju z bazy danych, Jeśli użytkownik jest zalogowany, tworzy odniesienie do dokumentu zawierającego informacje o nastroju w *kolekcji* na podstawie identyfikatora użytkownika. Po pomyślnym usunięciu, funkcja wywołuje procedurę odpowiedzialną pomyślne usunięcie, w przypadku błędu, funkcja wywołuje procedurę, odpowiedzialną za obsługę błędu.

Po etapach pobrania i zapisu danych, następuje analiza wpisów. Opracowana logika analizy nastrojów umożliwia poznanie dynamiki stanów emocjonalnych w wybranych okresach: rocznym, miesięcznym oraz tygodniowym. W ramach procesu analizy, prezentowane są średnie, minimalne i maksymalne wartości ocen nastroju, wraz z wykrytymi, najczęściej występującymi objawami w konkretnych okresach. Przy zastosowaniu dwóch typów wizualizacji: wykresu liniowego z uwzględnioną średnią wartością dla danej jednostki czasu oraz wykresu słupkowego przedstawiającego częstotliwość występowania poszczególnych ocen od 1 do 5. Do realizacji tego celu wykorzystano bibliotekę „*com.github.mikephil.charting”*, która jest stronniczym rozwiązaniem (*third party*).

Tabela 34. Listing kodu funkcji obsługującej pobranie rocznych wpisów do wykresu liniowego

```
fun getYearLineGraphEntry(
    successCallback: (List<Entry>) -> Unit,
    errorCallback: (String) -> Unit
) {
    dataManager.readMoodsLastYear(object : MoodsCallback {
        override fun onMoodsLoaded(moods: List<Mood>) {
            if (moods.isNotEmpty()) {
                val calendar = Calendar.getInstance()
                calendar.time = Date()
```

```
            calendar.add(Calendar.YEAR, -1)

            val entries = mutableListOf<Entry>()
            for (i in 0 until 12) {
               calendar.set(Calendar.DAY_OF_MONTH, 1)
               calendar.set(Calendar.HOUR_OF_DAY, 0)
               calendar.set(Calendar.MINUTE, 0)
               calendar.set(Calendar.SECOND, 0)
               calendar.set(Calendar.MILLISECOND, 0)
               val startDate = calendar.time
               calendar.add(Calendar.MONTH, 1)
               calendar.add(Calendar.DAY_OF_MONTH, -1)
               calendar.set(Calendar.HOUR_OF_DAY, 23)
               calendar.set(Calendar.MINUTE, 59)
               calendar.set(Calendar.SECOND, 59)
               calendar.set(Calendar.MILLISECOND, 999)
               val endDate = calendar.time

               val ratings = mutableListOf<Float>()
               var totalRating = 0f
               var totalDays = 0
               for (mood in moods) {
                  if (mood.createDate >= startDate && mood.createDate <= endDate) {
                     ratings.add(mood.rating.toFloat())
                     totalRating += mood.rating.toFloat()
                     totalDays++
                  }
               }
               val averageRating = if (totalDays != 0) totalRating / totalDays else 0f
               entries.add(Entry(i.toFloat(), averageRating))
               calendar.add(Calendar.MILLISECOND, 1)
               val monthLabel = SimpleDateFormat("MMM").format(startDate)
               val yearLabel = SimpleDateFormat("yyyy").format(startDate)
               entries.last().data = "$monthLabel $yearLabel"
            }
            successCallback(entries)
         }
      }

      override fun onError(error: ErrorCode) {
      }
   })
}
```

Źródło: Opracowanie własne, metoda getYearLineGraphEntry w widoku modelu StatisticViewModel

Powyższa fragment definicji funkcji znajdujący się w tabeli powyżej (patrz Tabela 34. Listing kodu funkcji obsługującej pobranie rocznych wpisów do wykresu liniowego), wykorzystuje „*readMoodsLastYear*” do odczytania listy nastrojów. Pobiera *instancje klasy Calendar* i przesuwa go o rok wstecz. *Iteruje* przez kolejne 12 miesięcy. Ustala datę początkową jako pierwszy dzień danego miesiąca, a datę końcową jako ostatni dzień. Przetwarza listę nastrojów, obliczając średnią ocenę nastroju

w danym okresie. Tworzy obiekt *Entry* zawierający numer miesiąca i średnią ocenę, a także datę jako etykietę.

Tabela 35. Listing kodu funkcji obsługującej pobranie miesięcznych wpisów do wykresu liniowego

```
fun getMonthLineGraphEntry(
    successCallback: (List<Entry>) -> Unit,
    errorCallback: (String) -> Unit
) {
    dataManager.readMoodsLastMonth(object : MoodsCallback {
        override fun onMoodsLoaded(moods: List<Mood>) {
            if (moods.isNotEmpty()) {
                val currentDate = Date()
                val calendar = Calendar.getInstance()
                calendar.time = currentDate
                calendar.add(Calendar.DAY_OF_MONTH, -30)

                val entries = mutableListOf<Entry>()
                for (i in 0 until 30) {
                    calendar.set(Calendar.HOUR_OF_DAY, 0)
                    calendar.set(Calendar.MINUTE, 0)
                    calendar.set(Calendar.SECOND, 0)
                    calendar.set(Calendar.MILLISECOND, 0)
                    val startDate = calendar.time
                    calendar.set(Calendar.HOUR_OF_DAY, 23)
                    calendar.set(Calendar.MINUTE, 59)
                    calendar.set(Calendar.SECOND, 59)
                    calendar.set(Calendar.MILLISECOND, 999)
                    val endDate = calendar.time

                    val ratings = mutableListOf<Float>()
                    var totalRating = 0f
                    var totalDays = 0
                    for (mood in moods) {
                        if (mood.createDate >= startDate && mood.createDate <= endDate) {
                            ratings.add(mood.rating.toFloat())
                            totalRating += mood.rating.toFloat()
                            totalDays++
                        }
                    }
                    val averageRating = if (totalDays != 0) totalRating / totalDays else 0f
                    entries.add(Entry(i.toFloat(), averageRating))
                    calendar.add(Calendar.MILLISECOND, 1)
                    val dayLabel = SimpleDateFormat("dd").format(startDate)
                    val monthLabel = SimpleDateFormat("MMM").format(startDate)
                    entries.last().data = "$dayLabel $monthLabel "
                }
                successCallback(entries)
            }
        }
        override fun onError(error: ErrorCode) {
        }
```

```
        }
)
}
```
Źródło: Opracowanie własne, metoda getMonthLineGraphEntry w widoku modelu StatisticViewModel

Przedstawiona funkcja z tabeli powyżej (Patrz Tabela 35. Listing kodu funkcji obsługującej pobranie miesięcznych wpisów do wykresu liniowego), jest niezbędna do generowania danych potrzebnych do utworzenia wykresu liniowego, który prezentuje średnie oceny nastrojów w przeciągu ostatnich 30 dni. Wykorzystuje ona repozytorium danych do odczytu listy nastrojów, następnie określa bieżącą datę i przesuwa kalendarz o 30 dni wstecz. W ciągu *iteracji* dla każdego wpisu z 30 dni ustala datę początkową jako pierwszą chwilę danego dnia, a datę końcową jako ostatnią chwilę tego samego dnia. Przetwarza listę nastrojów, obliczając średnią ocenę nastroju w danym okresie. Tworzy obiekt *Entry*, który zawiera numer dnia, obliczoną średnią ocenę nastroju oraz datę jako etykietę. Proces zakańcza się wywołując *„successCallback*" z gotową listą wpisów.

Tabela 36. Listing kodu obsługujący pobranie tygodniowych wpisów do wykresu liniowego

```
fun getWeekLineGraphEntry(
    successCallback: (List<Entry>) -> Unit,
    errorCallback: (String) -> Unit
) {
    dataManager.readMoodsLastWeek(object : MoodsCallback {
        override fun onMoodsLoaded(moods: List<Mood>) {
            if (moods.isNotEmpty()) {
                val currentDate = Date()
                val calendar = Calendar.getInstance()
                calendar.time = currentDate
                calendar.add(Calendar.DAY_OF_MONTH, -6)
                val entries = mutableListOf<Entry>()
                for (i in 0 until 7) {
                    calendar.set(Calendar.HOUR_OF_DAY, 0)
                    calendar.set(Calendar.MINUTE, 0)
                    calendar.set(Calendar.SECOND, 0)
                    calendar.set(Calendar.MILLISECOND, 0)
                    val startDate = calendar.time
                    calendar.set(Calendar.HOUR_OF_DAY, 23)
                    calendar.set(Calendar.MINUTE, 59)
                    calendar.set(Calendar.SECOND, 59)
                    calendar.set(Calendar.MILLISECOND, 999)
                    val endDate = calendar.time

                    val ratings = mutableListOf<Float>()
                    var totalRating = 0f
                    var totalDays = 0
                    for (mood in moods) {
                        if (mood.createDate >= startDate && mood.createDate <= endDate) {
                            ratings.add(mood.rating.toFloat())
```

```
                        totalRating += mood.rating.toFloat()
                        totalDays++
                    }
                }
                val averageRating = if (totalDays != 0) totalRating / totalDays else 0f
                entries.add(Entry(i.toFloat(), averageRating))
                calendar.add(Calendar.MILLISECOND, 1)
                val dayLabel = SimpleDateFormat("EEE dd").format(startDate)
                val monthLabel = SimpleDateFormat("MMM").format(startDate)
                entries.last().data = "$dayLabel $monthLabel"
            }
            successCallback(entries)
        }
    }

    override fun onError(error: ErrorCode) {
        //errorCallback(error)
    }
  })
}
```

Źródło: Opracowanie własne, metoda getWeekLineGraphEntry w widoku modelu StatisticViewModel

Powyższa definicja metody (patrz Tabela 36. Listing kodu obsługujący pobranie tygodniowych wpisów do wykresu liniowego), ma za zadanie generować dane niezbędne do stworzenia wykresu liniowego, prezentującego średnie oceny nastrojów w ciągu ostatnich 7 dni. Funkcja wykorzystuje *repozytorium danych* do odczytu listy nastrojów, następnie pobiera bieżącą datę i przesuwa kalendarz o 6 dni wstecz. Tworzy listę obiektów wpisów oraz iteruje przez kolejne 7 dni, ustala datę początkową jako północ danego dnia, a datę końcową jako ostatni moment tego samego dnia, następująco oblicza średnią ocenę nastroju w tym okresie na podstawie wczytanej listy nastrojów, ostatecznie wywołuje *„successCallback”* z gotową listą wpisów.

Tabela 37. Listing kodu obsługujący pobranie wpisów rocznych lub miesięcznych do wykresu słupkowego

```
fun getYearBarGraphEntry(
    successCallback: (List<BarEntry>) -> Unit
) {
    val dataManager= DataManager()
    dataManager.readMoodsLastYear(object : MoodsCallback {
        override fun onMoodsLoaded(moods: List<Mood>) {
            val ratings = moods.map { it.rating }
            val ratingCounts = IntArray(5)
            for (rating in ratings) {
                if (rating in 1..5) {
                    ratingCounts[rating - 1]++
                }
            }
```

```
            val barEntries = ratingCounts.mapIndexed { index, count ->
                BarEntry(index.toFloat() + 1, count.toFloat())
            }
            successCallback(barEntries)
        }
        override fun onError(errorCode: ErrorCode) {
        }})}
```

Źródło: Opracowanie własne, metoda getYearMonthBarGraphEntry w widoku modelu StatisticViewModel

Powyższa funkcja (patrz Tabela 37. Listing kodu obsługujący pobranie wpisów rocznych do wykresu słupkowego), ma na celu generowanie danych potrzebnych do stworzenia wykresu słupkowego reprezentującego liczbę wystąpień poszczególnych ocen nastrojów w ciągu roku. Metoda wykorzystuje repozytorium danych do odczytania listy nastrojów, na ich podstawie oblicza oceny i tworzy listę ich wystąpień, następnie tworzy tablicę zawierającą liczbę ocen, w której dla każdej oceny przechowuje liczbę jej wystąpień, po czym ma miejsce mapowanie „*ratingCounts*" do listy obiektów *BarEntry*, reprezentujących wartości na osi x oceny i wysokość słupka na osi y liczbę wystąpień. Po wykonaniu powyższych operacji, wywoływany jest „*successCallback*", przekazując gotową listę „*barEntries*".

Tabela 38. Listing kodu obsługujący pobranie tygodniowych wpisów do wykresu słupkowego

```
fun getMonthBarGraphEntry(
    successCallback: (List<BarEntry>) -> Unit
) {
    val dataManager= DataManager()
    dataManager.readMoodsLastMonth(object : MoodsCallback {
        override fun onMoodsLoaded(moods: List<Mood>) {
            val ratings = moods.map { it.rating }
            val ratingCounts = IntArray(5)
            for (rating in ratings) {
                if (rating in 1..5) {
                    ratingCounts[rating - 1]++
                }
            }
            val barEntries = ratingCounts.mapIndexed { index, count ->
                BarEntry(index.toFloat() + 1, count.toFloat())
            }
            successCallback(barEntries)
        }
        override fun onError(errorCode: ErrorCode) {
        }
    })
}
```

Źródło: Opracowanie własne

Powyższa funkcja w tabeli (patrz Tabela 38. Listing kodu obsługujący pobranie wpisów miesięcznych do wykresu słupkowego), odpowiedzialna jest za utworzenie danych

potrzebnych do stworzenia wykresu słupkowego reprezentującego liczbę wystąpień poszczególnych ocen nastrojów w ciągu ostatniego tygodnia.

Tabela 39. Listing kodu obsługujący pobranie tygodniowych wpisów do wykresu słupkowego

```
fun getWeekBarGraphEntry(
    successCallback: (List<Entry>) -> Unit
) {
    val dataManager= DataManager()
    dataManager.readMoodsLastWeek (object : MoodsCallback {
        override fun onMoodsLoaded(moods: List<Mood>) {
            val ratings = moods.map { it.rating }
            val ratingCounts = IntArray(5)
            for (rating in ratings) {
                if (rating in 1..5) {
                    ratingCounts[rating - 1]++
                }
            }
            val barEntries = ratingCounts.mapIndexed { index, count ->
                BarEntry(index.toFloat() + 1, count.toFloat())
            }
            successCallback(barEntries)
        }
        override fun onError(errorCode: ErrorCode) {
        }
    })
}
```

Źródło: Opracowanie własne

Powyższa funkcja w tabeli (patrz Tabela 39. Listing kodu obsługujący pobranie wpisów tygodniowych do wykresu słupkowego), służy do generowania danych potrzebnych do stworzenia wykresu słupkowego reprezentującego liczbę wystąpień poszczególnych ocen nastrojów w ciągu ostatniego tygodnia.

W powyższej aplikacji kluczowym elementem jest automatyczne wykrywanie zaburzeń odżywiania. Po dodaniu nowego rekordu do bazy, aplikacja analizuje dane z ostatniego miesiąca, identyfikując wzorce związane z zaburzeniami. Jeśli takie wzorce zostaną znalezione, użytkownik otrzymuje powiadomienie lub informację, co pozwala na świadome reagowanie na ewentualne problemy zdrowotne.

Celem tworzenia powiadomień jest informowanie użytkowników o ewentualnych sygnałach związanych z zaburzeniami odżywiania, co pozwala im na większą świadomość i reakcję w razie potrzeby. Ostateczne diagnozy i decyzje powinny być podejmowane we współpracy z profesjonalnymi specjalistami zdrowia.

Tabela 40. Listing kodu funkcji obsługującą wczesne wykrywanie zaburzeń odżywiania

```
fun analyzeMoods() {
      dataManager.readMoodsLastMonth(object : MoodsCallback {
         override fun onMoodsLoaded(moods: List<Mood>) {
            if (moods.isNotEmpty()) {
               val disorderCounts = analyzeSymptomsForEatingDisorders(moods)
               val totalDisorderCounts = sumUpDisorderCounts(disorderCounts)
               val disease = findMostCommonDisorder(totalDisorderCounts)
               when (disease) {
                  "Anorexia Nervosa" -> {
                     dataManager.readProfile(object : ProfileCallback {
                        override fun onProfileLoaded(profile: Profile?) {
                           if (profile != null) {
                              val bmi = calculateBMI(profile)
                              if (bmi < 18.5) {
                                 val notification= Notification(disease)
                                 dataManager.createNotification(notification)
                              }
                           }
                        }
                        override fun onError(error: ErrorCode) {
                        }
                     })
                  }
                  "Bulimia Nervosa" -> {
                     val symptomIdsToCheck = setOf(
                        "Consuming a large amount of food in a short period",
                        "Loss of control over the amount of food consumed"
                     )
                     val symptomCounts = mutableMapOf<String, Int>()
                     val oneMonthAgo =
LocalDate.now().minusMonths(1).atStartOfDay(ZoneId.systemDefault()).toInstant()
                     val oneMonthAgoDate = Date.from(oneMonthAgo)
                     for (mood in moods) {
                        if (mood.createDate >= oneMonthAgoDate) {
                           mood.list?.forEach { symptom ->
                              if (symptom.id in symptomIdsToCheck) {
                                 symptomCounts[symptom.id] = (symptomCounts[symptom.id] ?: 0) + 1
                              }
                           }
                        }
                     }
                     val weeklySymptoms = listOf(
                        "Consuming a large amount of food in a short period",
                        "Vomiting",
                        "Use of laxatives"
                     )
                     val recurringWeeklySymptoms = weeklySymptoms.any { symptomId ->
                        symptomCounts[symptomId] ?: 0 >= 4
                     }
                     if (recurringWeeklySymptoms) {
                        val notification= Notification(disease)
                        dataManager.createNotification(notification)
                     }
                  }
```

```
                    else -> {
                        val notification= Notification(disease)
                        dataManager.createNotification(notification)
                    }
                }
}}override fun onError(error: ErrorCode) {
}})}
```

Źródło: Opracowanie własne, fragment definicji klasy MoodAnalyze

Kod w powyższej tabeli (patrz Tabela 40. Listing kodu funkcji obsługującą wczesne wykrywanie zaburzeń odżywiania), służy do analizy nastrojów w kontekście potencjalnych zaburzeń odżywiania, jest to najważniejsza funkcjonalność w autorskiej aplikacji Mood Tracker.

Po wczytaniu nastrojów z okresu ostatniego miesiąca, użyto metody „*analyzeSymptomsForEatingDisorders*", opisanej w Tabeli 41 (patrz Listing kodu funkcji obsługującej analizę symptomów). Ta funkcja jest odpowiedzialna za analizę występujących objawów i związanych z nimi zaburzeń. Następnie, wywoływana jest procedura „*sumUpDisorderCounts*" (patrz Tabela 42. Listing kodu funkcji obsługującej zliczanie ilości symptomów), agreguje ona liczbę wystąpień poszczególnych zaburzeń. W zależności od najczęściej występującego zaburzenia, funkcja wykonuje różne akcje:

- Dla anoreksji psychicznej, wczytuje profil użytkownika i oblicza jego wskaźnik BMI. Jeśli jest on mniejszy niż 18.5, tworzy powiadomienie o diagnozie.
- Dla Bulimii psychicznej analizuje konkretne symptomy z ostatniego miesiąca, używając „*symptomIdsToCheck*". Liczy, ile razy każdy z tych objawów wystąpił w ostatnim miesiącu. Sprawdza, czy objawy, które powinny występować co najmniej raz w tygodniu (4 razy w miesiącu), spełniają ten warunek.
- W innych przypadkach, tworzy powiadomienie o najczęściej występującym zaburzeniu.

Tabela 41. Listing kodu funkcji obsługującej analizę symptomów

```
private fun analyzeSymptomsForEatingDisorders(moods: List<Mood>): Map<String, Int> {
    val symptomMapping = mapOf(
        "Extreme restriction of food intake unrelated to illness" to listOf("Anorexia Nervosa"),
        "Thinking about food, weight, and body appearance" to listOf("Anorexia Nervosa"),
        "Unrealistic perception of own body" to listOf("Anorexia Nervosa","Bulimia Nervosa"),
        "Consuming a large amount of food in a short period" to listOf("Binge Eating
Disorder","Bulimia Nervosa"),
        "Vomiting" to listOf("Bulimia Nervosa","Drunkorexia","Anorexia Nervosa"),
        "Use of laxatives" to listOf("Bulimia Nervosa","Drunkorexia"),
        "Taking actions to revert to previous weight" to listOf("Anorexia Nervosa","Bulimia
Nervosa","Drunkorexia"),
```

```
        "Doing everything to avoid gaining weight" to listOf("Anorexia Nervosa","Bulimia
Nervosa","Drunkorexia"),
        "Stress related to participating in social eating experiences" to listOf("Avoidant/Restrictive
Food Intake Disorder (ARFID)"),
        "Loss of control over the amount of food consumed" to listOf("Binge Eating
Disorder","Bulimia Nervosa" ),
        "Strong feeling of shame after a large meal" to listOf("Binge Eating Disorder"),
        "Frequent checking of body weight in a short period" to listOf("Bulimia Nervosa"),
        "Checking body measurements using a measuring tape or mirror reflections" to
listOf("Anorexia Nervosa"),
        "Avoiding certain food groups or categories" to listOf("Avoidant/Restrictive Food Intake
Disorder (ARFID)"),
        "Consuming substances of low nutritional value or unusual items" to listOf("Pica"),
        "Difficulty swallowing food unrelated to illness" to listOf("Avoidant/Restrictive Food Intake
Disorder (ARFID)"),
        "Feeling of blockage in the throat while eating unrelated to illness" to
listOf("Avoidant/Restrictive Food Intake Disorder (ARFID)"),
        "Coughing or choking during meals unrelated to illness" to listOf("Avoidant/Restrictive
Food Intake Disorder (ARFID)"),
        "Restricting meals to drink alcohol" to listOf("Drunkorexia")
    )
    val disorderCounts = mutableMapOf<String, Int>()
    for (mood in moods) {
        val symptoms = mood.list ?: emptyList()
        for (symptom in symptoms) {
            val disorders = symptomMapping[symptom.id]
            if (disorders != null) {
                for (disorder in disorders) {
                    disorderCounts[disorder] = (disorderCounts[disorder] ?: 0) + 1
                }
            }
        }
    }
    return disorderCounts
}
```

Źródło: Opracowanie własne, fragment metody definicja klasy NotificationViewModel

Procedura zawarta w tabeli (patrz Tabela 41. Listing kodu funkcji obsługującej analizę symptomów) przetwarza listę nastrojów i analizuje występujące w nich symptomy, aby określić, które zaburzenia żywieniowe mogą być związane z danymi objawami. Wykorzystuje mapowanie symptomy-zaburzenia, które określa, które zaburzenia mogą być związane z danym objawem. Dla każdego nastroju następuje *iteracje* przez symptomy, sprawdza się związane z nimi zaburzenia z mapowania, a następnie zwiększa licznik wystąpień tych zaburzeń. Wynik to *mapa*, gdzie *klucze* to nazwy zaburzeń, a wartości to liczba ich wystąpień w analizowanej grupie nastrojów. Metoda ta pozwala uzyskać skuteczną analizę symptomów w grupie nastrojów, określa potencjalne związki między występującymi objawami a zaburzeniami odżywiania.

Tabela 42. Listing kodu funkcji obsługującej zliczanie ilości symptomów

```
private fun sumUpDisorderCounts(disorderCounts: Map<String, Int>): Map<String, Int> {
    val totalDisorderCounts = mutableMapOf<String, Int>()
    for ((disorder, count) in disorderCounts) {
        val disorderName = disorder.substringBefore("(").trim()
        totalDisorderCounts[disorderName] = (totalDisorderCounts[disorderName] ?: 0) + count
    }
    return totalDisorderCounts }
```

Źródło: Opracowanie własne, fragment metody definicja klasy NotificationViewModel

Funkcja umieszczona w tabeli powyżej (patrz Tabela 42. Listing kodu funkcji odpowiedzialny za wyszukanie zaburzenia) jest istotna, ponieważ umożliwia przypisanie symptomów pasujących do kilku zaburzeń odżywiania oraz jednoczesne określenie wskaźnika liczby przesłanek dla konkretnego zaburzenia odżywiania.

Tabela 43. Listing kodu funkcji odpowiedzialny za wyszukanie zaburzenia

```
private fun findMostCommonDisorder(disorderCounts: Map<String, Int>): String {
    return disorderCounts.entries.maxByOrNull { it.value }?.key ?: ""
}
```

Źródło: Opracowanie własne, fragment metody definicja klasy NotificationViewModel

Zawarta w powyższej tabeli (patrz Tabela 43. Listing kodu funkcji odpowiedzialny za wyszukanie zaburzenia) definicja metody pozwala z pośród zmapowanych liczebności „przesłanek" dla potencjalnego występowania danego zaburzenia, określić których z nich jest najwięcej, finalnie zwracając nazwę danego zaburzenia. Funkcja ta jest kluczowa, ponieważ jednoznacznie określa zaburzenie odżywiania, które może zostać wcześnie wykryte.

Tabela 44. Listing kodu funkcji wyliczającej wskaźnik BMI

```
private fun calculateBMI(profile: Profile): Double {
    val height = profile.height
    val weight = profile.weight
    if (height > 0 && weight > 0) {
        val heightInMeters = height.toDouble() / 100.0
        return weight.toDouble() / (heightInMeters * heightInMeters)
    }
    return 0.0
}
```

Źródło: Opracowanie własne

W tabeli powyżej (patrz Tabela 44. Listing kodu funkcji wyliczającej wskaźnik BMI) przedstawiono metodę pozwalającą obliczyć wskaźnik masy ciała (BMI) na podstawie wartości wzrostu i wagi profilu użytkownika. Kluczowym aspektem metody jest fakt, że pozwala na jednoznaczną identyfikacje, czy występujące symptomy mogą zostać sklasyfikowane, jako wskaźnik do wczesnego wykrycia i dalszego ostrzegania przed anoreksją psychiczną,

co wynika z punktu 1 w tabeli (patrz Tabela 3. Anoreksja - kryteria diagnostyczne), opisującej kryteria diagnostyczne dla powyższego zaburzenia.

Tabela 45. Listing kodu funkcji obsługującej zapis rekordu offline

```
private var firestore: FirebaseFirestore
init {
    val firestoreSettings = FirebaseFirestoreSettings.Builder()
        .setPersistenceEnabled(true)
        .build()
    firestore = FirebaseFirestore.getInstance()
    firestore.firestoreSettings = firestoreSettings
}
```

Źródło: Opracowanie własne, fragment definicji klasy DataManager

W powyższej tabeli (patrz Tabela 45. Listing kodu funkcji obsługującej zapis rekordu offline), decydujący dla tego kodu jest fakt, że inicjalizuje on instancję bazy danych Firestore z włączoną obsługą pracy w trybie offline oraz wykorzystaniem *puli połączeń*. Funkcjonalność ta umożliwia aplikacji działanie bez dostępu do Internetu. Wykorzystanie *puli połączeń* Zapewnia optymalne zarządzanie połączeniami do bazy danych. W trybie offline dane są synchronizowane z serwerem po ponownym nawiązaniu połączenia, co zapewnia spójność i niezawodność działania aplikacji.

Tabela 46. Listing kodu klasy enumerowanej ErrorCode

```
enum class ErrorCode{  EMR, EMA, EMC, EMU, EMAU, EMC15, EMD , EPR , EPU ,}
```

Źródło: Opracowanie własne

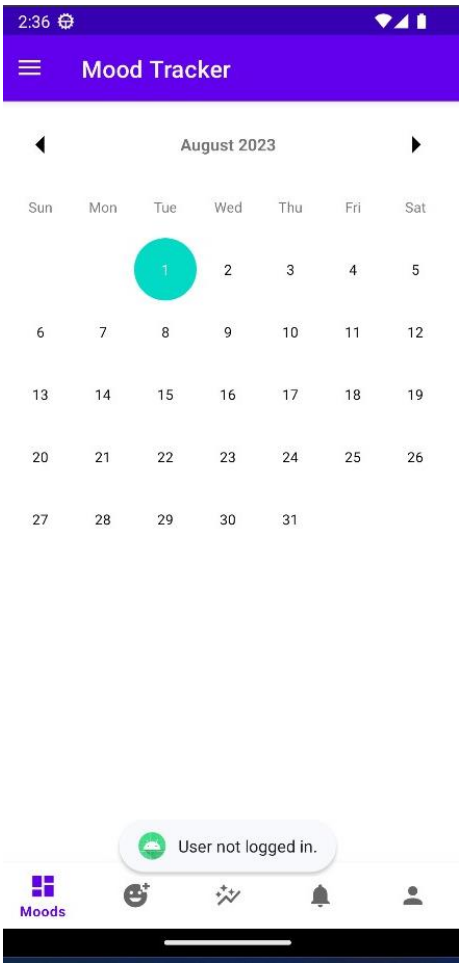
Kod w powyższej tabeli (patrz Tabela 46. Listing kodu klasy enumerowanej ErrorCode), jest ważny ponieważ odpowiada za uproszczenie oraz ujednolicenie mnogości błędów związanych z bazą danych.

Objaśnienie elementów klasy:

- „*EMA*" - Błąd *autentykacji* nastrojów;
- „*EMC*" - Błąd tworzenia nastroju;
- „*EMU*" - Błąd aktualizacji nastroju;
- „*EMAU*" - Błąd *autentykacji* przy aktualizacji nastroju;
- „*EMC15*" - Błąd tworzenia nastroju – w ciągu ostatnich 15 minut, nastrój został już utworzony;
- *„EMD*" - Błąd usuwania nastroju;
- „*EPR*" - Błąd odczytu profilu;
- „*EPU*" - Błąd aktualizacji profilu;

3.3. Instrukcja użytkowania aplikacji

Rysunek 3. Prezentacja interfejsu użytkownika

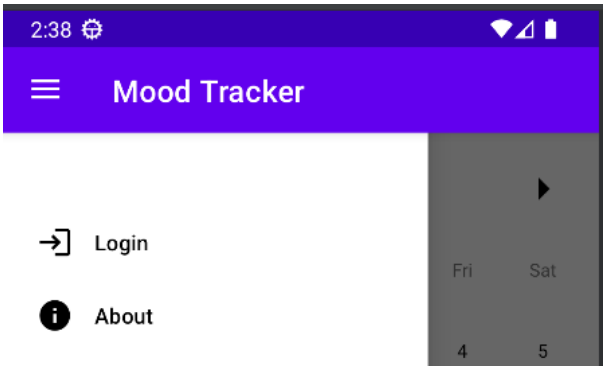


Źródło: Opracowanie własne

Na rysunku (patrz Rysunek 3. Prezentacja interfejsu użytkownika). przedstawiono interfejs graficzny aplikacji, w nim znajduje się menu górne oraz dolne, które umożliwiają szybki dostęp do różnych funkcji aplikacji. Menu górne zawiera kluczowe opcje dotyczące operacji logowania i informacji o aplikacji, natomiast menu dolne zawiera 5 fragmentów, które pozwalają na swobodne poruszanie się po aplikacji.

W głównej części ekranu znajduje się interaktywny kalendarz, umożliwia on użytkownikowi łatwe i szybkie przeglądanie różnych dni. Dzięki temu użytkownik może w prosty sposób nawigować do konkretnego dnia, który chce przeanalizować. Kalendarz jest zintegrowany z funkcją rejestrowania nastrojów, co umożliwia użytkownikowi zobaczenie, jak zmieniały się jego emocje w różnych dniach.

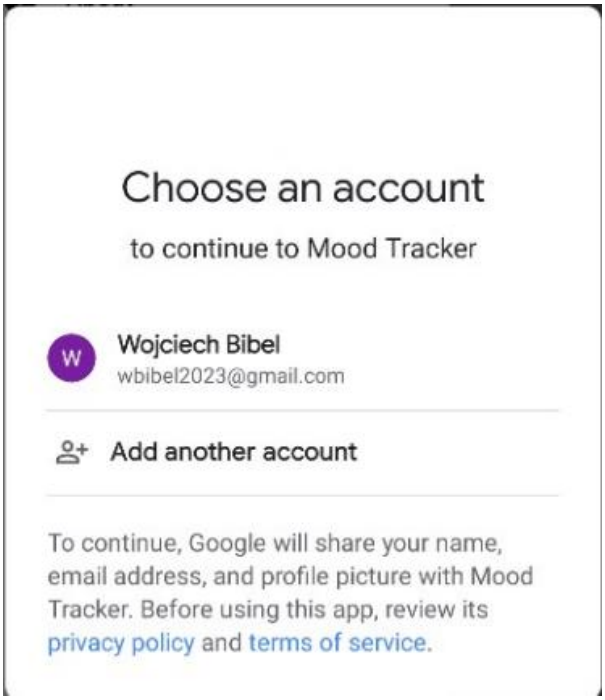
Rysunek 4. Prezentacja menu górnego



Źródło: Opracowanie własne

Powyższy rysunek (patrz Rysunek 4. Prezentacja menu górnego), przedstawia dostępne opcje w menu górnym, przed zalogowaniem. Opcja login wywołuje proces logowania, natomiast opcja o aplikacji (*about*) wywołuje aktywność, która opisuje powyższą aplikację.

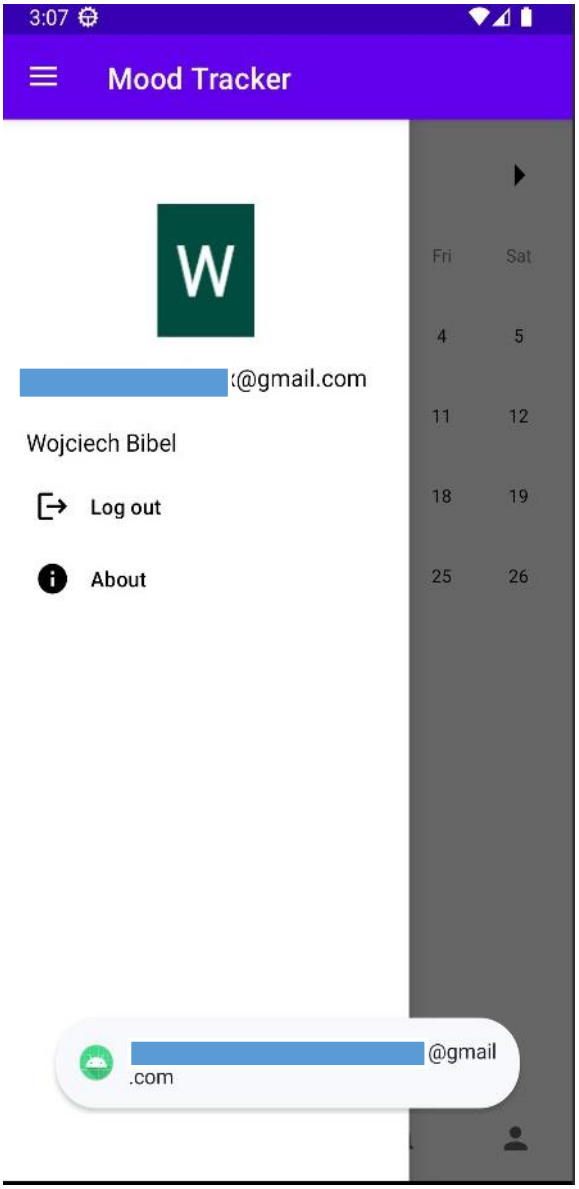
Rysunek 5. Logowanie przez Google



Źródło: Opracowanie własne

Na rysunku (patrz Rysunek 5. Logowanie przez Google), przedstawiono efekt naciśnięcia na opcje zaloguj. Pojawia się logowanie przy użyciu konta Google, pozwala to na logowanie przy użyciu istniejącego konta, co pozwala na użycie takich zabezpieczeń jak, pytania bezpieczeństwa, potwierdzenie sms.

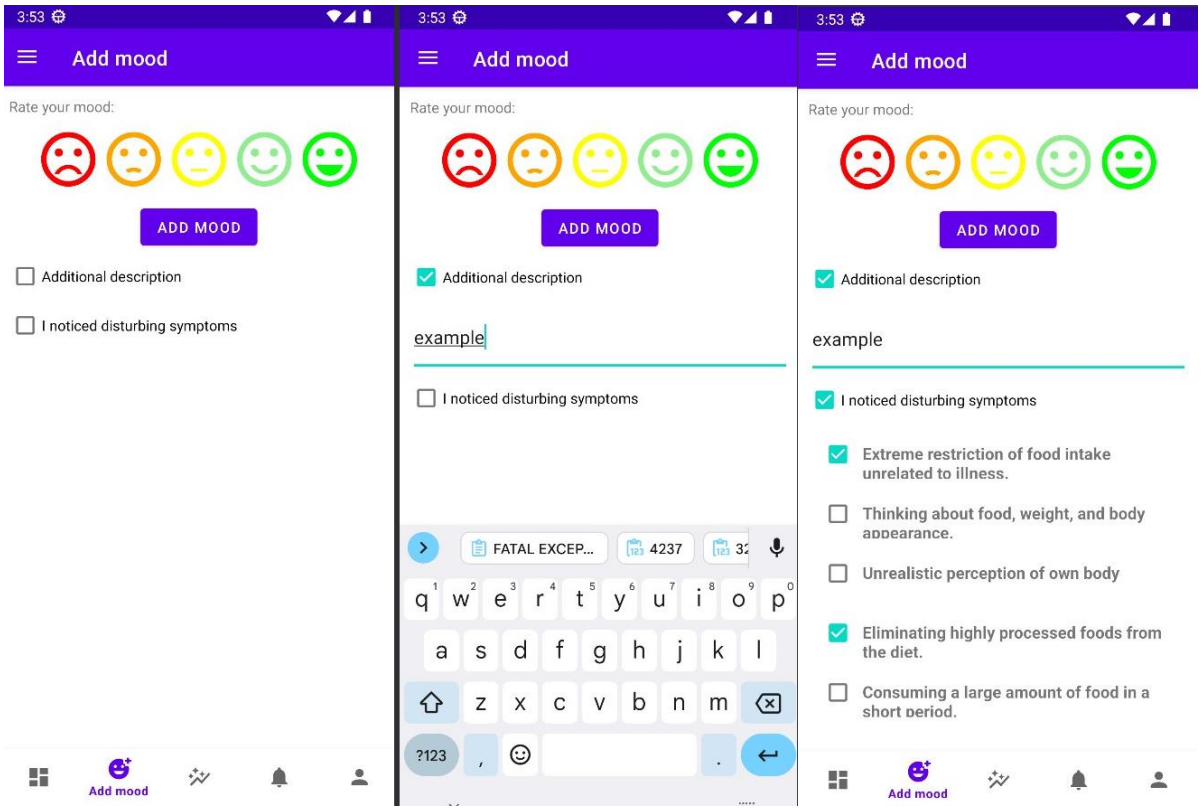
Rysunek 6. Prezentacja po zalogowaniu



Źródło: Opracowanie własne

Niniejszy rysunek (patrz Rysunek 6. Prezentacja po zalogowaniu), przedstawia aktualizacje interfejsu użytkownika po zalogowaniu.

Rysunek 7. Prezentacja dodawania rekordu



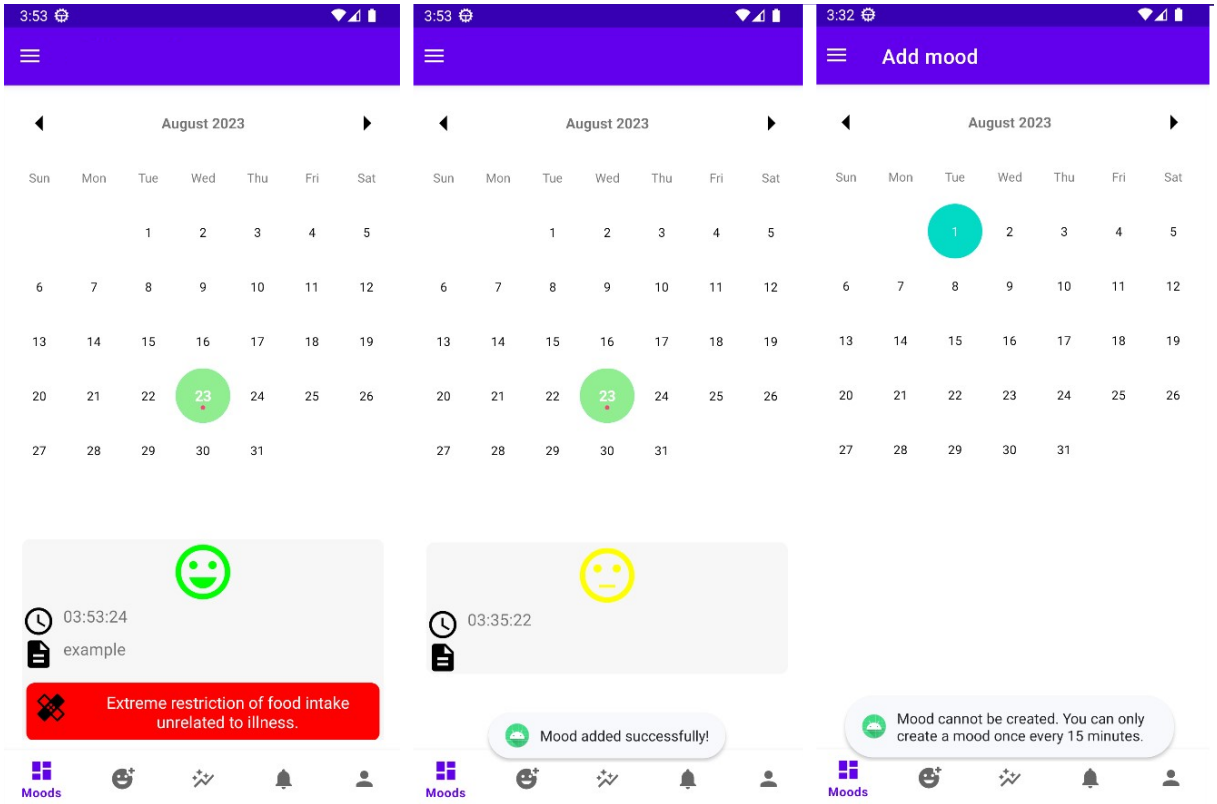
Źródło: Opracowanie własne

Powyższy Rysunek (patrz Rysunek 7. Prezentacja dodawania rekordu), przedstawia fragment dodania nastroju, określenia swojego samopoczucia poprzez wybór odpowiedniej emoji oraz dodania opcji objawów zaburzeń odżywiania. Proces ten jest zaprojektowany w sposób intuicyjny i łatwy, aby użytkownik mógł szybko i dokładnie zarejestrować swoje doświadczenia.

Użytkownik może wybrać ikonę emoji, która najlepiej oddaje jego obecny nastrój. To pozwala na szybkie i precyzyjne wyrażenie swojego stanu emocjonalnego. Jednocześnie użytkownik ma opcję dodania informacji dotyczących ewentualnych objawów zaburzeń odżywiania. Może to obejmować zarówno opis tekstowy (nie brany pod uwagę w analizie), jak i wybór z listy dostępnych objawów.

Po zakończeniu wprowadzania danych użytkownik może zapisać rekord, a wszystkie informacje zostaną zapisane w bazie danych. To umożliwia późniejsze przeglądanie wpisów, analizowanie trendów nastrojów oraz ewentualnych zmian w nawykach żywieniowych. Cały proces ma na celu dostarczenie użytkownikowi narzędzia do lepszego zrozumienia swojego stanu emocjonalnego oraz zdrowia w kontekście odżywiania.

Rysunek 8. Prezentacja po dodaniu rekordu



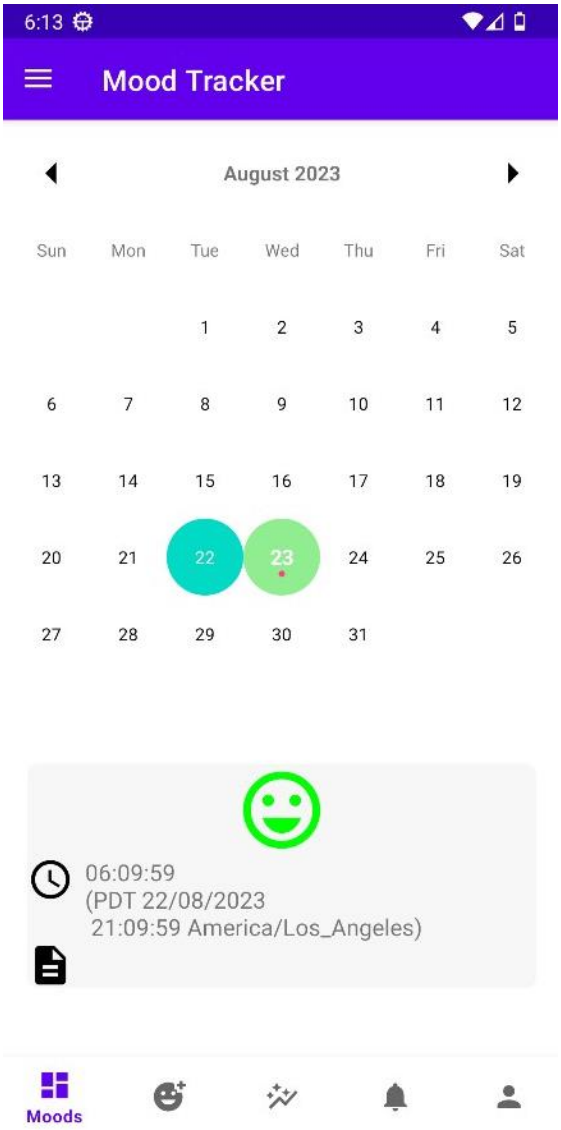
Źródło: Opracowanie własne

Powyższy Rysunek (patrz Rysunek 8. Prezentacja po dodaniu rekordu), przedstawia rezultaty operacji dodania nastroju. Po dodaniu rekordu, system wyświetla komunikat potwierdzający powodzenie lub brak dodania wpisu. Ten komunikat informuje użytkownika, czy operacja zakończyła się.

Dodatkowo, po dodaniu rekordu, dekorator kalendarza zostaje zaktualizowany. Wizualna reprezentacja kalendarza zostaje zmodyfikowana, aby odzwierciedlać nowe informacje. Dla konkretnego dnia, dla którego został dodany rekord, zostaje wyliczona średnia wartość nastroju na podstawie wprowadzonych emocji. To pozwala użytkownikowi szybko ocenić, jakie były jego nastroje w danym dniu.

Pod kalendarzem, po dodaniu rekordu, pojawia się nowy wpis. Ten wpis zawiera informacje o dacie, wybranym nastroju w postaci emoji oraz dodatkowe objawy zaburzeń odżywiania, jeśli zostały wprowadzone. To umożliwia użytkownikowi szybkie przeglądanie swoich wpisów, analizowanie zmian w nastrojach.

Rysunek 9. Obsługa stref czasowych



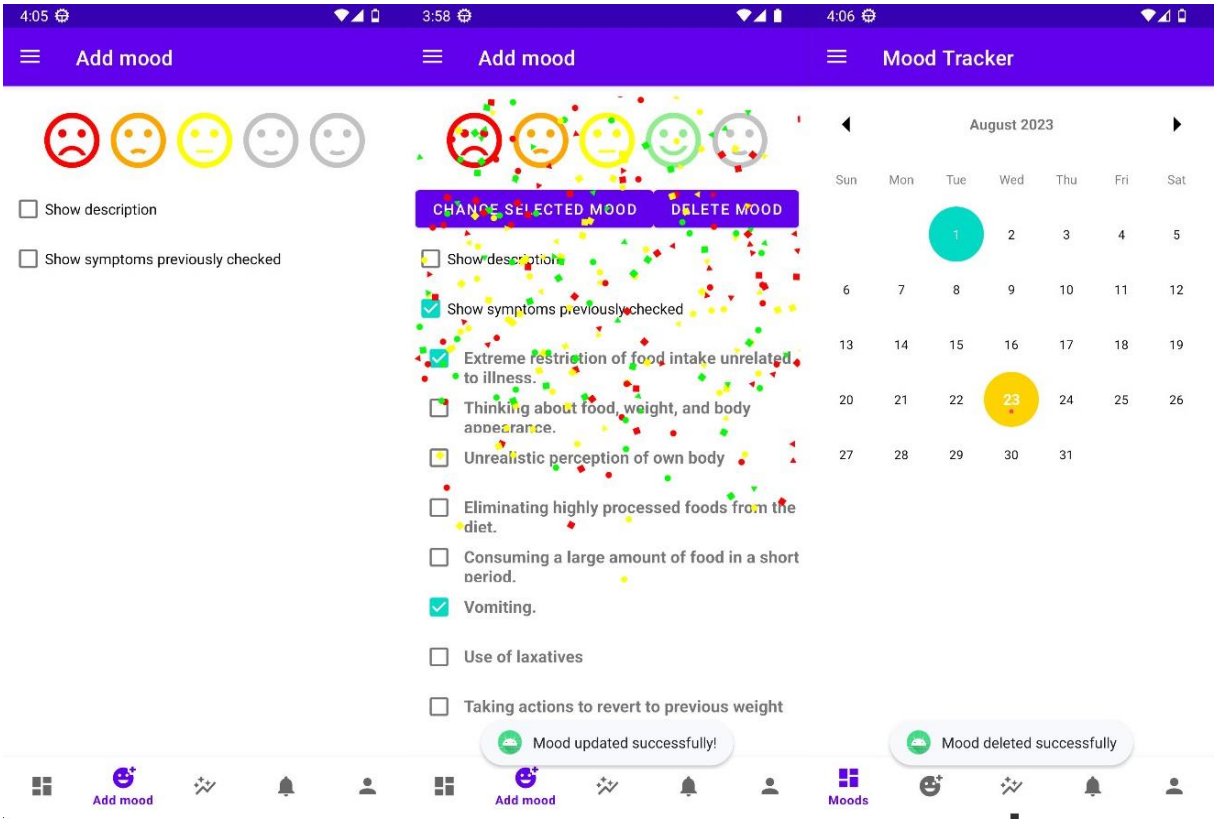
Źródło: Opracowanie własne

Na rysunku (patrz Rysunek 9. Obsługa stref czasowych), przedstawiono zmiany przy obsłudze innych stref czasowych niż tej w której znajduje się użytkownik. Kiedy użytkownik przemieszcza się do innej strefy czasowej, aplikacja automatycznie dostosowuje czas w sczytanych wpisach, tak aby były odzwierciedleniem obecnej lokalnej daty godziny. To gwarantuje dokładność danych i zapobiega zamieszaniu związanemu ze zmianą strefy czasowej.

Po dodaniu rekordu, użytkownik ma 15 minut na edycję lub usunięcie wpisu. Aby przejść do edycji lub usunięcia, należy nacisnąć na rekord znajdujący się pod kalendarzem. Jeśli wystąpiła jakaś pomyłka lub zmiana zdania, użytkownik może wtedy zmodyfikować

dane rekordu lub go usunąć. Po upływie tego czasu, użytkownik może jedynie obejrzeć rekord, lecz nie będzie miał możliwości dokonania edycji ani usunięcia. To ograniczenie czasowe zapewnia kontrolę nad wprowadzonymi danymi, jednocześnie zabezpieczając przed przypadkowymi lub nieautoryzowanymi zmianami w historii.

Rysunek 10. Sczytanie pojedynczego rekordu, edycja, usunięcie



Źródło: Opracowanie własne

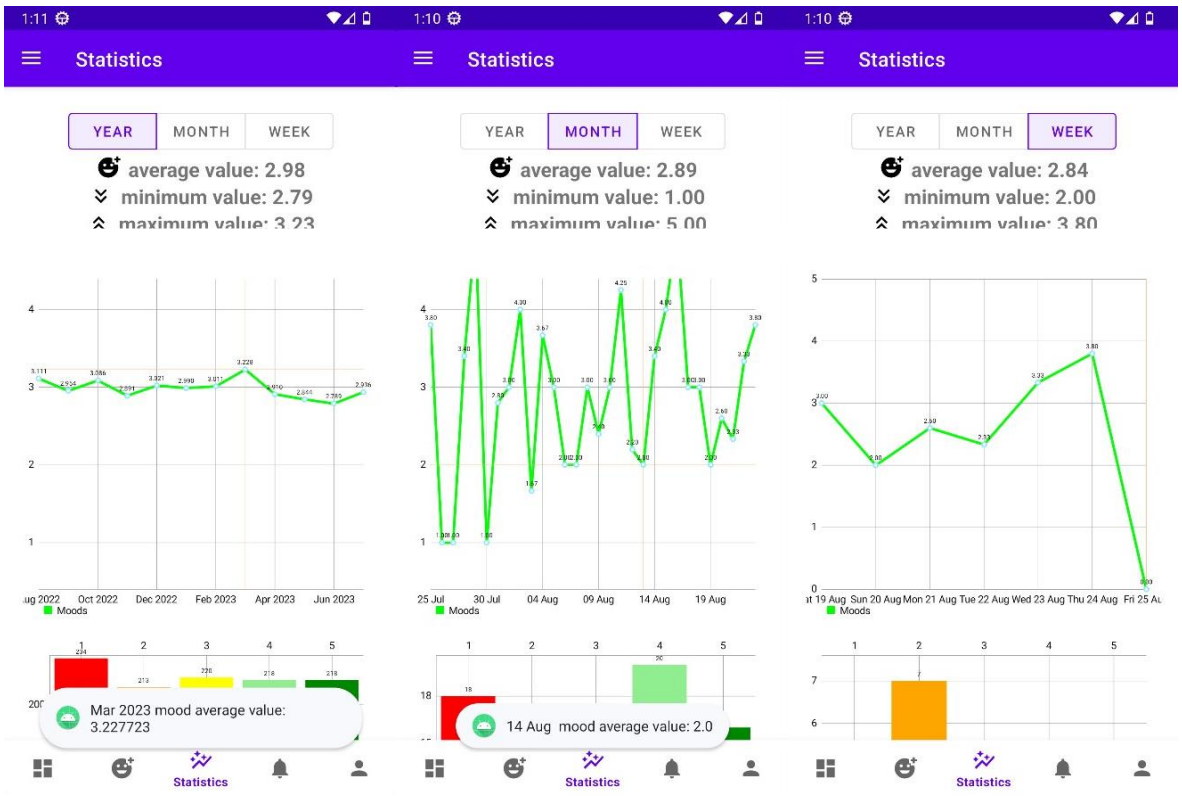
Powyższy Rysunek (patrz Rysunek 10. Sczytanie pojedynczego rekordu, edycja, usunięcie), przedstawia rezultaty operacji odczytu, aktualizacji, usunięcia nastroju. Jeśli rekord jest młodszy niż 15 minut, użytkownik ma możliwość dokonania operacji na nim - może go zmienić lub usunąć. Natomiast, jeśli rekord jest starszy niż 15 minut, użytkownik nie ma już możliwości wprowadzania zmian w nim.

Po pomyślnej operacji na rekordzie, tzn. po zmianie lub usunięciu, pojawia się efekt wizualny w postaci konfetti. Ten efekt wizualny ma na celu dostarczenie użytkownikowi przyjemnej i satysfakcjonującej informacji zwrotnej, że operacja została zakończona sukcesem. To dodatkowy element interakcji, który może sprawić, że korzystanie z aplikacji będzie bardziej angażujące dla użytkownika.

Po dodaniu rekordów, użytkownik ma możliwość analizy swoich nastrojów. Opracowana logika analizy nastrojów umożliwia zrozumienie zmian w stanach emocjonalnych w wybranych okresach: rocznym, miesięcznym oraz tygodniowym. W ramach tego procesu, aplikacja przedstawia charakterystyki średniej, minimalnej i maksymalnej oceny nastroju w tych konkretnych okresach. Dla lepszego zrozumienia danych, dostępne są dwie typy wizualizacji.

Wykres liniowy na rysunku poniżej (patrz Rysunek 11. Prezentacja wykresu liniowego), przedstawia graficznie zmiany nastroju w czasie. Każda wybrana jednostka czasu (rok, miesiąc, tydzień) posiada własną linię na wykresie. Ta linia reprezentuje średnią wartość nastroju w danym okresie. Dzięki temu użytkownik może w prosty sposób zidentyfikować trendy i ewentualne zmiany w swoich emocjach na przestrzeni czasu. Powyższy komponent, umożliwia także przybliżanie wartości dla dokładniejszej analizy krótszych okresów. Po kliknięciu na konkretny punkt na wykresie, użytkownikowi wyświetlana jest szczegółowa informacja o dokładnej wartości nastroju w wybranej jednostce czasu. To pozwala użytkownikowi na bardziej dogłębną analizę swojego stanu emocjonalnego w konkretnych momentach.

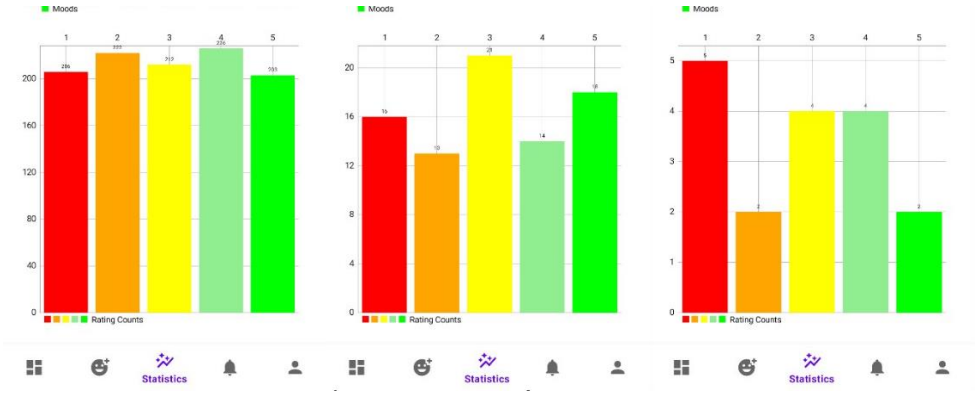
Rysunek 11. Prezentacja wykresu liniowego



Źródło: Opracowanie własne

Wykres na rysunku poniżej (patrz Rysunek 12. Prezentacja wykresu słupkowego), przedstawia częstotliwość występowania poszczególnych ocen nastroju w skali od 1 do 5. To umożliwia użytkownikowi zobaczenie, jak często występowały różne nastroje i oceny. Przy naciśnięciu na punkt na wykresie, pojawia się szczegółowa informacja o dacie i szczegółach z danego nastroju. Powyższe wykresy są przybliżane, co pozwala na dokładniejsze spojrzenie na wybrany okres czasu.

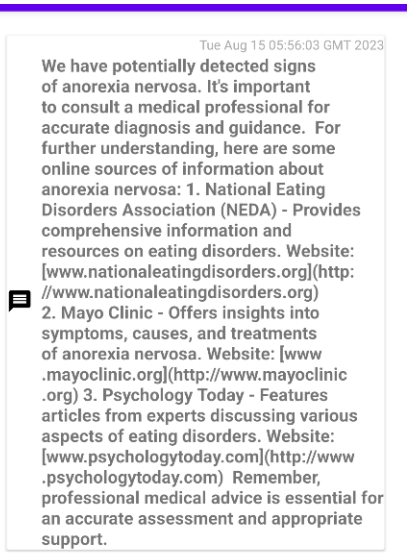
Rysunek 12. Prezentacja wykresu słupkowego



Źródło: Opracowanie własne

Przedstawione wizualizacje pomagają użytkownikowi spojrzeć na swoje rekordy z perspektywy czasu i uzyskać głębsze zrozumienie swojego stanu emocjonalnego w różnych okresach. To dodatkowa funkcja, która wspiera użytkownika w samopoznaniu i świadomości własnych nastrojów.

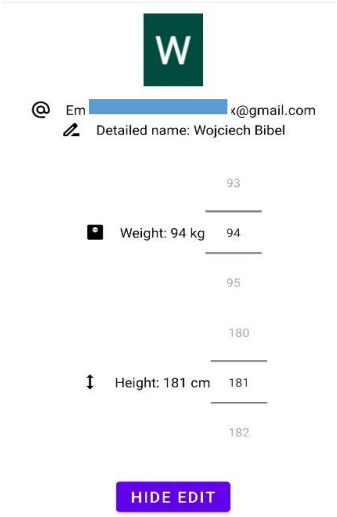
Rysunek 13. Prezentacja fragmentu powiadomień



Źródło: Opracowanie własne

Powyższy Rysunek (patrz Rysunek 13. Prezentacja fragmentu powiadomień), pokazuje powiadomienie w formie ostrzeżenia.

Rysunek 14. Prezentacja edycji profilu



Źródło: Opracowanie własne

Powyżej na rysunku (patrz Rysunek 14. Prezentacja edycji profilu), przedstawiono fragment odpowiadający za personalizacje profilu, posiada on możliwość edycji wagi i wzrostu oraz opisuje zalogowanego użytkownika.

W celu dokładniejszego rozróżnienia zaburzeń odżywiania, takich jak bulimia czy anoreksja, konieczne jest wprowadzenie informacji dotyczących wagi i wzrostu. Te dane są istotne, aby aplikacja mogła lepiej ocenić ewentualne nieprawidłowości w nawykach żywieniowych. Dodatkowo, aplikacja korzysta z informacji związanych z kontem email oraz nazwą pobraną z konta Google, aby personalizować doświadczenie użytkownika.

Ważne jest jednak zrozumienie, że analiza nie zastępuje kontaktu z lekarzem. To narzędzie umożliwia wczesne wykrywanie na podstawie dostępnej literatury przedmiotu, ale może nie być idealne w identyfikacji skomplikowanych zaburzeń, ponieważ jest dziedzina wiedzy będą w stałym rozwoju. Dlatego też, użytkownik powinien pamiętać, że najlepszym podejściem jest skonsultowanie się z profesjonalistą zdrowia, jeśli pojawią się jakiekolwiek obawy związane z zaburzeniami odżywiania.

Po analizie każdego dodanego rekordu, aplikacja może wyświetlić ostrzeżenia dotyczące potencjalnych zaburzeń odżywiania. Te ostrzeżenia mają na celu zwrócenie uwagi użytkownika na ewentualne nieprawidłowości i zachęcenie do dalszej refleksji oraz ewentualnego kontaktu ze specjalistą. To dodatkowe zabezpieczenie, które ma na celu

zapewnić użytkownikowi bezpieczeństwo i świadomość związane z ich zdrowiem psychicznym i fizycznym.

Rysunek 15. Język polski

Źródło: Opracowanie własne

Na powyższym rysunku (patrz Rysunek 15. Język polski), przedstawiono działanie tłumaczonych zasobów, elementy interfejsu graficznego są tłumaczone na język polski.

3.4. Analiza SWOT

Mocne Strony (*Strengths*):

- Personalizacja: Aplikacja wykorzystuje informacje z konta Google, dostosowując doświadczenie użytkownika.
- Analiza Nastrojów: Możliwość śledzenia i analizy nastrojów w różnych okresach, dzięki czemu użytkownik zyskuje lepsze zrozumienie swojego stanu emocjonalnego.
- Wykrywanie Zaburzeń Odżywiania: Funkcja analizy może wczesnym etapie wykrywać potencjalne zaburzenia odżywiania i wyświetlać ostrzeżenia.
- Wizualizacje: Wykresy liniowe i słupkowe dostarczają wizualnej reprezentacji danych, ułatwiającej analizę i zrozumienie dynamiki zmian nastrojów w okresach czasowych.

Słabe Strony (*Weaknesses*):

- Niezawodność Analizy: Pomimo korzystania z dostępnej literatury, analiza może nie zawsze dokładnie wykrywać wszystkie rodzaje zaburzeń odżywiania.
- Brak Profesjonalnej Diagnozy: Aplikacja nie zastępuje kontaktu z lekarzem w przypadku poważnych problemów zdrowotnych.
- Ograniczenia Firecloud: Aplikacja korzysta z usługi Firecloud, która ma ograniczenia na ilość odczytów i zapisów danych w wersji bezpłatnej (np. 50 000 odczytów i 10 000 zapisów). Te ograniczenia mogą wpływać na skalowalność i wydajność aplikacji, szczególnie w środowisku produkcyjnym, gdzie liczba użytkowników i operacji może znacznie wzrosnąć. Wymienione limity mogą być niewystarczające do obsługi dużej ilości użytkowników, co potencjalnie prowadzi do ograniczenia dostępności aplikacji lub konieczności przeprowadzania płatnych uaktualnień, aby zwiększyć dostępne zasoby.
- Kompatybilność z Urządzeniami: Aplikacja była testowana na ograniczonej liczbie urządzeń, co może nie wystarczać dla mnogości typów ekranów dostępnych na rynku. Różnice w specyfikacjach, rozmiarach ekranów i sposobie obsługi różnych urządzeń mogą wpłynąć na jakość interfejsu użytkownika oraz ogólną użyteczność aplikacji. Konieczne jest ciągłe dostosowywanie interfejsu użytkownika do różnych urządzeń, aby zapewnić spójność i zadowolenie użytkowników.

Szanse (*Opportunities*):

- Edukacja i Świadomość: Aplikacja może pomóc użytkownikom zrozumieć swoje nastroje i nawyki żywieniowe, co przyczynia się do lepszej samoświadomości.

- Rozwój Funkcji: Aplikacja może stale ulepszać analizę i wizualizacje, dostarczając użytkownikom bardziej precyzyjne informacje.

Zagrożenia (*Threats*):

- Prywatność Danych: Ze względu na dostęp do danych osobowych, konieczna jest troska o ich bezpieczeństwo i ochronę przed ewentualnymi naruszeniami.
- Dostępność Profesjonalnej Pomocy: Użytkownicy mogą polegać na aplikacji zamiast szukać pomocy u specjalistów, co może opóźnić właściwe leczenie.
- Ograniczenia Firecloud: Ograniczenia w Firecloud mogą wpłynąć na wydajność i skalowalność aplikacji, zwłaszcza jeśli liczba użytkowników wzrośnie.
- Struktura Wywołań Zwrotnych: Przy dalszym rozwoju aplikacji przy może stanowić problem, utrudniając zrozumienie i utrzymanie czystości kodu.

ZAKOŃCZENIE

Przedstawiona praca licencjacka jest strukturalnie podzielona na rozdziały, które poświęcone są kolejno zaburzeniom odżywiania, analizie rynku aplikacji mobilnych oraz omówieniu stworzonej aplikacji.

Celem utworzenia autorskiej aplikacji jest uświadamianie użytkowników o zaburzeniach odżywiania, zwłaszcza tych, którzy mogą o nich nie wiedzieć. Powyższy cel może zostać jedynie osiągnięty częściowo. Skuteczność aplikacji tego rodzaju, jest częściowo zależna od czynnika ludzkiego i podejścia, jakie użytkownicy przyjmą wobec tej aplikacji. Powyższe rozwiązanie dostarcza cenne informacje, ostatecznie korzyści i efekty zależą od zaangażowania i świadomości samych użytkowników.

Powyższa autorska aplikacja Mood Tracker, posiada wiele mocnych stron. Jedną z nich jest monitorowanie i analiza nastrojów, która daje możliwość wczesnego wykrywania zaburzeń odżywiania, ta funkcjonalność przyczynia się do zapobiegania powstawaniu zaburzeń odżywiania. Wczesne wykrycie problemu pozwoli na szybszą interwencję za pomocą skutecznych metod leczenia zaburzeń odżywiania. Aplikacja ma potencjał do edukacji i wsparcia w samoświadomości jej użytkowników. Jednakże, jej ograniczeniem jest niezawodność analizy, brak profesjonalnej diagnozy oraz ograniczenia wynikające z używania darmowej wersji usług Firecloud. Wartością dodaną przygotowanej aplikacji jest możliwość wizualizacji.

BIBLIOGRAFIA

KSIĄŻKI

1. Brytek-Matera, A., *Obraz ciała - obraz siebie, Wizerunek własnego ciała w ujęciu społecznym*, Wydawnictwo Difin, Warszawa 2008

2. Brytek-Matera, A., *Zaburzenia odżywiania*, Wydawnictwo Lekarskie PZWL, Warszawa 2022

3. Fairburn, C.G., *Terapia poznawczo-behawioralna i zaburzenia odżywiania*, Wydawnictwo Uniwersytetu Jagielońskiego, Kraków 2013

4. Gaber, K., Kuk, B., *Kiedy cierpisz na zaburzenia odżywiania*, Wydawnictwo Lekarskie PZWL, Warszawa 2022

5. Lock, J., Grange, D., *Twoje dziecko i zaburzenia odżywiania. Jak mu pomóc?*, Wydawnictwo Znak, Kraków 2006

6. Mroczkowska, C., Ziółkowska, B., Cwojdzińska, A., *Zaburzenia odżywiania poradnik dla rodziców i bliskich*, Wydawnictwo Naukowe Scholar, Warszawa 2007

7. Wycisk, J., Ziółkowska, B., *Młodzież przeciwko sobie Zaburzenia odżywiania i samouszkodzenia – jak pomóc nastolatkom w szkole*, Wydawnictwo Difin, Warszawa 2010

CZASOPISMA

1. Bąk-Sosnowska, M., *Differential criteria for binge eating disorder and food addictionin the context of causes and treatment of obesity*, „Psychiatria Polska", nr. 51, 2017

2. Devoe, D., Han, A., Anderson, A., Katzman, D.K., Patten, S.B., Soumbasis, A., Flanagan, J., Paslakis, G., Vyver, E., Marcoux, G., Dimitropoulos, G., *The impact of the COVID -19 pandemic on eating disorders*: A systematic review, „International Journal of Eating Disorders", nr. 1, 2022

3. Fairburn, C.G., Bohn, K., *Eating disorder NOS (EDNOS): an example of the troublesome "not otherwise specified" (NOS) category in DSM-IV*, „ Behaviour Research and Therapy'', nr. 43, 2005

4. Giacomo, E., Aliberti, F., Pescatore, F., Santorelli, M., Pessina, R., Placenti, V., Colmegna, F., Clerici, M., *Disentangling binge eating disorder and food addiction: a systematic review and meta-analysis*, „ Eating and Weight Disorders - Studies on Anorexia, Bulimia and Obesity", nr. 27, 2022

6. Hay, P., *Current approach to eating disorders: a clinical update*, „ Internal Medicine Journal", nr. 50, 2020

7. Jowik, K., Dutkiewicz, A., Słopień, A., Tyszkiewicz-Nwafor, M., *A multi-perspective analysis of dissemination, etiology, clinical view and therapeutic approach for binge eating disorder (BED)*, „Psychiatria Polska", 54, 2020

8. Levinson, C.A., Fewell, L., Brosof, L.C., *My Fitness Pal calorie tracker usage in the eating disorders*, „Eating Behaviors", nr 27, 2017

9. Li, S., Cui, G., Yin, Y., Tang, K., Chen, L., Liu, X, *Prospective Association Between Problematic Mobile Phone Use and Eating Disorder Symptoms and the Mediating Effect of Resilience in Chinese College Students: A 1-Year Longitudinal Study*, „Frontiers in Public Health", nr 10, 2022

10. Michalska, A., Szejko, N., Jakubczyk, A., Wojnar, M., *Nonspecific eating disorders – a subjective review*, „Psychiatria Polska'', nr.50 ,2016

11. Rikani, A.A., Choudhry, Z., Choudhry, A.M., Ikram, H., Asghar, M.W., Kajal, D., Waheed, A., Mobassarah N.J., *A critique of the literature on etiology of eating disorders*, „Annals of Neurosciences", 20, 2013

12. Schueller, S.M., Neary, M., Lai, J., Epstein, D.A., *Understanding People's Use of and Perspectives on Mood Tracking Apps: An Interview Study (Preprint)*, „JMIR Mental Health", 2021

13. Szynal, K., Górski, M., Grajek, M., Ciechowska, K., Polaniak, R., *Drunkorexia – knowledge review*, „Psychiatria Polska", nr. 56 , 2022

14. van Eeden, A. E., D. van Hoeken, Hoek, H.W., *Incidence, prevalence and mortality of anorexia nervosa and bulimia nervosa*, „Current Opinion in Psychiatry", nr. 1, 2021

STRONY INTERNETOWE

1. https://play.google.com/store/search?q=eating%20disorder&c=apps&hl=pl, dostęp: 05.02.2023

2. https://icd.who.int, dostęp: 05.02.2023

3. https://www.android.com/intl/pl_pl/everyone/facts/, dostęp: 19.08.2023

SPIS TABEL

SPIS RYSUNKÓW