

ივანე ჯავახიშვილის სახელობის თბილისის სახელმწიფო უნივერსიტეტი

ზუსტ და საბუნებისმეტყველო მეცნიერებათა ფაკულტეტი

ჯგუფური პროექტი

**ფინანსების მართვის აპლიკაცია**

საბაკალავრო პროგრამა - კომპიუტერული მეცნიერება

პროექტის შემსრულებლები: **ნიჟარაძე საბა,**

**ჭავჭანიძე თამთა,**

**ბარბაქაძე პაატა,**

**წიკლაური ნიკოლოზი**

ხელმძღვანელი: ხაჩიძე დავით, პროფ.

ოდიშელიძე ნანა, ასისტ. პროფ.

თბილისი

2025

ანოტაცია

პროექტის მიზანია ინოვაციური მობილური აპლიკაციის შექმნა, რომელიც მომხმარებლებს დაეხმარება საკუთარი ფინანსების ეფექტურად მართვაში. იგი პასუხობს ისეთ პრობლემებს, როგორიცაა ზედმეტი ხარჯვა, დანაზოგების არარსებობა და ფინანსური მიზნების გაურკვევლობა. არსებული ფინანსური აპლიკაციებისგან განსხვავებით, ეს პლატფორმა სთავაზობს არა მხოლოდ მონაცემთა სტატისტიკურ ანალიზს, არამედ პერსონალიზებულ რჩევებსა და სტრატეგიულ დაგეგმარებას.

აპლიკაცია ეფუძნება მომხმარებლის ფინანსური ქცევის ანალიზს და ამ მონაცემებზე დაყრდნობით აწვდის ინდივიდუალურ რეკომენდაციებს. გადაწყვეტილებების მიღება ხორციელდება ალგორითმებზე დაფუძნებული სისტემით, რომელიც იყენებს როგორც ფიქსირებულ წესებს, ასევე პერსონალიზებული რეკომენდაციების ალგორითმებს. მომავალში შესაძლებელია მანქანური სწავლის ინტეგრაცია.

პროექტის ტექნიკური შესრულება მოიცავს:

• მობილური აპლიკაციის შექმნას React Native-ით [4]

• სერვერის ნაწილი Nest.js [3]

• მონაცემთა შენახვას PostgreSQL-ში [1]

• ავტორიზაციის სისტემას JWT (JSON Web Token) პროტოკოლით

• პერსონალიზებული რჩევებისთვის მონაცემთა ანალიზს

ლიტერატურა

[1] PostgreSQL Documentation – <https://www.postgresql.org/docs>

[2] Eric Freeman, Elisabeth Robson – Head First Design Patterns

[3] NestJS Documentation - <https://docs.nestjs.com/>

[4] React Native Documentation - <https://reactnative.dev/docs/getting-started>

Abstract

**ქართული ანოტაციის თარგმანი**

The project aims to develop an innovative mobile application that helps users manage their personal finances more effectively. It addresses common challenges such as overspending, lack of savings, and unclear financial goals. Unlike existing financial apps that focus mainly on static data visualization, this application provides dynamic analysis and personalized recommendations.

The solution is based on analyzing the user’s financial behavior and delivering individualized advice tailored to their habits and needs. The recommendation system relies on algorithm-based logic, including rule-based and dynamic personalized suggestions. In the future, machine learning models may be integrated for deeper insights.

The technical implementation includes:

• Mobile development using React Native [4]

• Backend using Nest.js [3]

• Data storage with PostgreSQL [1]

• Authentication with JWT (JSON Web Token) Protocol

• Data analysis and personalized suggestions

References

[1] PostgreSQL Documentation – <https://www.postgresql.org/docs>

[2] Eric Freeman, Elisabeth Robson – Head First Design Patterns

[3] NestJS Documentation - <https://docs.nestjs.com/>

[4] React Native Documentation - https://reactnative.dev/docs/getting-started

# სარჩევი

[სარჩევი 3](#_Toc137124259)

[შესავალი 4](#_Toc137124260)

[პროექტის მიზანი 4](#_Toc137124261)

[ამოცანები 4](#_Toc137124262)

[განხორციელების გზები 5](#_Toc137124263)

[პროექტის შესრულების განრიგი 5](#_Toc137124264)

[პროექტის ფარგლებში შექმნილი პროგრამული უზრუნველყოფის კონცეპტუალური მოდელი 6](#_Toc137124265)

[პროგრამული სისტემის ფუნქციონირების აღწერა 7](#_Toc137124266)

[დასკვნა 7](#_Toc137124267)

[ლიტერატურა 7](#_Toc137124268)

# შესავალი

დღესდღეობით, ადამიანებისთვის ფინანსების მართვა ერთ-ერთ მნიშვნელოვან გამოწვევად იქცა.მიუხედავად იმისა, რომ ბაზარზე არსებობს მრავალი ფინანსური აპლიკაცია, მათ შორის მობილ ბანკები,როგორიცაა მაგალითად საქართველოს ბანკისა და თიბისი ბანკის აპლიკაციები ,ისინი მომხმარებლებს საკუთარ დანახარჯებზე ინფორმაციას აწვდიან და ეს პროდუქტები ხშირად მხოლოდ მონაცემთა სტატიკურ ვიზუალიზაციაზეა ორიენტირებული. ისინი იშვიათად სთავაზობენ მომხმარებლებს კონკრეტულ რჩევებს ან გამოსავალს, რომელიც მათ ფინანსურ მდგომარეობას გააუმჯობესებს.

ფინანსების ეფექტური მართვა მოითხოვს არა მხოლოდ ხარჯების კონტროლსა და დაზოგვას, არამედ სტრატეგიულ დაგეგმარებას და ანალიტიკურ ხედვას. ადამიანები ყოველდღიურად აწყდებიან ისეთ პრობლემებს, როგორებიცაა თანხების გადაჭარბებული ხარჯვა, დანაზოგების არარსებობა ან ფინანსური მიზნების გაურკვევლობა. აღნიშნული გამოწვევების ფონზე, საჭიროა ინსტრუმენტი, რომელიც შეძლებს მომხმარებლების ფინანსური ქცევის სრულყოფილ ანალიზს და მათი საჭიროებების შესაბამისად ინდივიდუალური რჩევების შეთავაზებას.

ამ საჭიროებებიდან გამომდინარე, ჩვენ ვხედავთ შესაძლებლობას შევქმნათ ინოვაციური აპლიკაცია, რომელიც მომხმარებლებს არა მხოლოდ ხარჯების აღრიცხვის საშუალებას მისცემს, არამედ დინამიკურ ანალიზსა და პერსონალიზებულ რეკომენდაციებსაც შესთავაზებს. ჩვენი პროდუქტი დაეხმარება მომხმარებლებს ფინანსური მიზნების განსაზღვრაში, ხარჯების ოპტიმიზაციაში და უკეთესი ფინანსური გადაწყვეტილებების მიღებაში, რაც საბოლოოდ გაზრდის მათ ფინანსურ სტაბილურობას და კეთილდღეობას. აპლიკაცია ასევე მიაწვდის მომხმარებლებს ინფორმაციას ბაზარზე არსებულ ფინანსურ ტენდენციებზე, რაც მათ გადაწყვეტილებების მიღებაში დაეხმარება.

# 

# პროექტის მიზანი

ჩვენი აპლიკაციის მიზანია მომხმარებლებს დავეხმაროთ მათი ფინანსების ეფექტურად მართვაში, რის შედეგადაც ისინი მიიღებენ უკეთეს ფინანსურ გადაწყვეტილებებს და შეძლებენ უკეთესად მართონ მათი ფინანსური რესურსები. ჩვენი პროდუქტი მიზნად ისახავს შემდეგ პუნქტებს:

1. **ხარჯების ეფექტური მონიტორინგი:** აპლიკაცია საშუალებას მისცემს მომხმარებლებს გაანალიზონ მათი ყოველდღიური ხარჯები და შეიმუშაონ ბიუჯეტი, რაც ხელს შეუწყობს მათ ფინანსების ოპტიმიზაციას.
2. **პერსონალიზებული ფინანსური რჩევები:** აპლიკაცია უზრუნველყოფს ინდივიდუალურ რჩევებს და რეკომენდაციებს, რომლებიც მომხმარებლის ფინანსურ ქცევაზე და საჭიროებებზე იქნება მორგებული.
3. **დაზოგვების და ინვესტირების სტრატეგიები:** მომხმარებელს შეეძლება შეიმუშაოს სამომავლო მიზნების მიღწევის სტრატეგიები, მაგალითად, დანაზოგების შექმნა ან ინვესტირებისთვის საჭირო ნაბიჯების გადადგმა.

**ფინანსური მიზნების განსაზღვრა:** აპლიკაცია დაეხმარება მომხმარებლებს, რომ შეიმუშაონ კონკრეტული ფინანსური მიზნები, რაც მათ უკეთ დაეხმარება გეგმების მიღწევაში.

# ამოცანები

1. **ინტერფეისის შექმნა**  
   1.1. აპლიკაციის მომხმარებლის ინტერფეისის დიზაინის შექმნა  
   1.2. UI/UX დიზაინის პრინციპების გათვალისწინება   
   1.3. ინტერფეისის იმპლემენტაცია და ოპტიმიზაცია  
   1.4. აპლიკაციის ინტერფეისის ტესტირება (გატესტვა სხვადასხვა მოწყობილობებზე და ეკრანის ზომებზე)
2. **ფინანსური ანალიტიკის სისტემა**  
   2.1. ფინანსური მონაცემების შეგროვების და ორგანიზების სისტემის შექმნა  
   2.2. მონაცემთა ბაზის სტრუქტურის შემუშავება და ოპტიმიზაცია  
   2.3. სისტემის ინტეგრაცია ან ინტეგრაციისთვის მზაობა სხვადასხვა ფინანსურ წყაროებთან (ბანკები, საფინანსო სერვისები და სხვა)  
   2.4. სისტემის ფუნქციონალური ტესტირება
3. **პერსონალიზირებული რჩევების გაწევა**  
   3.1. მომხმარებლის ფინანსური ქცევის ანალიზი და შესაბამისი რეკომენდაციების ჩამოყალიბება  
   3.2. ალგორითმების შექმნა, რომლებიც უზრუნველყოფენ ინდივიდუალურ რჩევებს  
   3.3. პერსონალიზებული რეკომენდაციების მუდმივი ოპტიმიზაცია
4. **მონაცემთა ბაზების შექმნა**  
   4.1. მონაცემთა ბაზის შექმნისთვის საჭირო ტექნიკური მოთხოვნების განსაზღვრა  
   4.2. მონაცემთა სქემის და ბაზების ორგანიზების დაგეგმვა  
   4.3. ბაზების მიგრაციის და იმპლემენტაციის შესრულება  
   4.4. მონაცემთა კონსისტენციის და უსაფრთხოების უზრუნველყოფა (ავტორიზაცია / ავთენთიფიკაცია)  
   4.5. მონაცემთა ბაზის ტესტირება
5. **მობაილ აპლიკაციის შექმნა**5.1 ჩვენ მიერ შემუშავებული დიზაინის გადატანა მობაილ აპლიკაციაში  
   5.2. მობაილ აპლიკაციის ფუნქციონალური მახასიათებლების გაშვება (რეგისტრაცია, პროფილის შექმნა და სხვ.)  
   5.3. ბაზების ინტეგრაცია საიტთან
6. **პროგრამული უზრუნველყოფის ტესტირება და დებაგი**  
   6.1. აპლიკაციის სრულყოფილი ტესტირების გეგმის შექმნა  
   6.2. ფუნქციონალური და სრული ტესტირების ჩატარება  
   6.3. სისტემაში არსებული ხარვეზების გამოსწორება და სტაბილურობის გაუმჯობესება  
   6.4. აპლიკაციის საბოლოო ვერსიის ტესტირება  
   6.5. ტესტირების შედეგების დოკუმენტირება

# განხორციელების გზები

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | ამოცანა | განხორციელების გზა | შემსრულებლები | შედეგი |
| 1 | **ინტერფეისის დიზაინის შექმნა** | დიზაინის შემუშავება | ნიკოლოზ წიკლაური | შემუშავებული იქნება აპლიკაციის დიზაინი |
| 2 | **ინტერფეისის დიზაინის ოპტიმიზაცია** | ყოველი იტერაციის შემდეგ ტესტირება | ჯგუფის ყველა წევრი | მივიღებთ საბოლოო დიზაინს |
| 3 | **ფინანსური მონაცემების შეგროვება** | სხვადასხვა წყაროდან მონაცემების შეგროვება | პაატა ბარბაქაძე | მიღებული ფინანსური მონაცემები, შემდგომი დამუშავებისათვის |
| 4 | ფინანსური მონაცემების ინტეგრაცია | შეგროვებული მონაცემების სისტემასთან სწორად ინტეგრირება | ნიკოლოზ წიკლაური, პაატა ბარბაქაძე | მონაცემების განთავსება აპლიკაციაში შესაბამისი სტრუქტურით |
| 5 | ფინანსური მონაცემების ვიზუალიზაცია/ანალიტიკა | წარმოდგენილი მონაცემების დამუშავება ვიზუალურად და ანალიზი | ჯგუფის ყველა წევრი | მიღებული მონაცემების მიხედვით შექმნილი ფინანსური ვიზუალიზაცია და ანალიზი |
| 6 | ფინანსური მონაცემების ვიზუალიზაციის და ანალიტიკის ტესტირება | შესაბამისი ხელსაწყოებით და მიდგომებით მიღებული კომპონენტის ტესტირება | თამთა ჭავჭანიძე | გამართული ფინანსური ანალიზის ფუნქციონალი, მინიმალური ან 0 ხარვეზით |
| 7 | ფინანსური ქცევების ანალიზი პერსონალიზირებული რჩევებისთვის | მონაცემების დამუშავება იმგვარად რომ მოხდეს პერსონალური ქცევის ანალიზი | ჯგუფის ყველა წევრი | ანალიზის შედეგები პერსონალური რჩევებისთვის |
| 8 | პერსონალიზებული რეკომენდაციების მუდმივი ოპტიმიზაცია | რეკომენდაციების ანალიზი განგრძობითად | ჯგუფის ყველა წევრი | სრულყოფილი პერსონალური რჩევების სისტემა |
| 9 | მონაცემთა ბაზების ტექნოლოგიის შერჩევა და სტრუქტურის შექმნა | კონკრეტული მონაცემთა ბაზების სისტემის შექმნა და სტრუქტურიზება | საბა ნიჟარაძე, პაატა ბარბაქაძე | მონაცემთა ბაზის პირველადი სტრუქტურის მიღება |
| 10 | შექმნილი მონაცემთა ბაზის ტესტირება | ბაზის კომპონენტების შესაბამისად გატესტვა და ანალიზი | თამთა ჭავჭანიძე | მზა მონაცემთა ბაზა აპლიკაციასთან ინტეგრაციისთვის |
| 11 | აპლიკაციის შექმნა, მზა დიზაინის იმპლემენტაცია და სატესტო გარემოში განთავსება | თავდაპირველი აპლიკაციის შექმნა, დიზაინის იმპლემენტირება და TestFlight-ზე განთავსება | ნიკოლოზ წიკლაური | არაფუნქციონალური ხელშესახები პროდუქტის მიღება |
| 12 | ბექენდ სისტემის შექმნა და აპლიკაციაში ინტეგრირება | ფუნქციონალის შესაბამისი ბექენდ სისტემის იმპლემენტირება NestJS-ზე | პაატა ბარბაქაძე | მობილური აპლიკაცია,რომელშიც ხელშესახებია ვიზუალური და ფუნქციური მხარეები |
| 13 | აპლიკაციის განახლებებისა და უწყვეტი ხელმისაწვდომობის უზრუნველყოფა | შესაბამისი ხელსაწყოებით და პლატფორმებით აპლიკაციის მუდმივი მდგრადობა | საბა ნიჟარაძე | მობილური აპლიკაციის განახლება და ხელმისაწვდომობა ხდება მარტივად და ეფექტურად |
| 14 | აპლიკაციის სრულყოფილი ტესტირების გეგმის შექმნა | შესაბამისი მიდგომებით აპლიკაციის ტესტირების გეგმის ჩამოყალიბება | თამთა ჭავჭანიძე | მიღებული სტანდარტიზებული გეგმა,რომლითაც ეფექტურად მოხდება აპლიკაციის მუდმივი ტესტირება |
| 15 | აპლიკაციის სრულყოფილი ტესტირება | გეგმის მიხედვით აპლიკაციის მუდმივი ტესტირება | თამთა ჭავჭანიძე | არსებული ხარვეზების დადგენა და ჩამოყალიბება |
| 16 | ტესტირებისას მიღებული ხარვეზების გამოსწორება | თითოეული ხარვეზის აღმოფხვრა შესაბაიმისი კომპეტენციის მიხედვით | ჯგუფის ყველა წევრი | პროდუქტში აღმოფხვრილი ხარვეზები და ოპტიმიზაცია |
| 17 | პროდუქტის ხარისხის დადგენა და შედეგების დოკუმენტირება | საბოლოო პროდუქტის ხარისხის განსაძღვრა შესაბამისი გეგმითა და ხელსაწყოებით, დასკვნის შემუშავება | თამთა ჭავჭანიძე | პროდუქტი მზადაა და ხელმისაწვდომია მომხმარებლებისთვის |

# პროექტის შესრულების განრიგი

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| დაგეგმილიაქტივობები |  | **I** თვე | |  |  | **II** თვე | |  |  | **III**  თვე | |  | **IV**  თვე | | | |
| I kv | II kv | III kv | IV kv | I kv | II kv | III kv | IV kv | I kv | II kv | III kv | IV kv | I kv | II kv | III kv | IV kv |
| **ინტერფეისის შექმნა** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| დიზაინის შემუშავება |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ყოველი იტერაციის შემდეგ ტესტირება |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **ფინანსური მონაცემების შეგროვება/დამუშავება** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| სხვადასხვა წყაროდან მონაცემების შეგროვება |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| შეგროვებული მონაცემების სისტემასთან სწორად ინტეგრირება |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| წარმოდგენილი მონაცემების დამუშავება ვიზუალურად და ანალიზი |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| მონაცემების დამუშავება იმგვარად რომ მოხდეს პერსონალური ქცევის ანალიზი |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **მონაცემთა ბაზები** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| კონკრეტული მონაცემთა ბაზების სისტემის შექმნა და სტრუქტურიზება |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ბაზის კომპონენტების შესაბამისად გატესტვა და ანალიზი |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| მობილური აპლიკაციის შექმნა |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| თავდაპირველი აპლიკაციის შექმნა, დიზაინის იმპლემენტირება და TestFlight-ზე განთავსება |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ფუნქციონალის შესაბამისი ბექენდ სისტემის იმპლემენტირება NestJS-ზე |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| შესაბამისი ხელსაწყოებით და პლატფორმებით აპლიკაციის მუდმივი მდგრადობა |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| პროგრამული უზრუნველყოფის ტესტირება |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| შესაბამისი მიდგომებით აპლიკაციის ტესტირების გეგმის ჩამოყალიბება |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| გეგმის მიხედვით აპლიკაციის მუდმივი ტესტირება |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| პროდუქტის ხარისხის დადგენა და შედეგების დოკუმენტირება |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

# პროექტის ფარგლებში შექმნილი პროგრამული უზრუნველყოფის კონცეპტუალური მოდელი

პროგრამული უზრუნველყოფა შედგება რამდენიმე ძირითადი ფუნქციური მოდულისგან, რომლებიც ერთობლივად უზრუნველყოფენ აპლიკაციის სრულყოფილ მუშაობას. ყოველი მოდული ერთმანეთთან დაკავშირებულია და რეალურ დროში ახორციელებს მონაცემთა გადაცემასა და მიღებას.

1. ინტერფეისის მოდული – რეალიზებულია React Native ტექნოლოგიით. უზრუნველყოფს მომხმარებლისთვის ინფორმაციის ვიზუალურ წარმოდგენას და ინტერაქციას სისტემასთან. აქ განხორციელებულია რეგისტრაცია, ავტორიზაცია, ხარჯების შეყვანა, ფინანსური მიზნების განსაზღვრა და პერსონალიზებული რჩევების ნახვა.

2. სერვერული ლოგიკის მოდული – შექმნილია Nest.js ჩარჩოზე. ის წარმოადგენს შუამავალს მომხმარებლის მოთხოვნებსა და მონაცემთა ბაზას შორის. შეიცავს ავტორიზაციის ლოგიკას (JWT), მონაცემთა დამუშავებასა და ანალიზის მექანიზმებს.

3. მონაცემთა ბაზის მოდული – გამოყენებულია PostgreSQL. ინახავს მომხმარებლის პროფილს, ფინანსურ ტრანზაქციებს, მიზნებს და რეკომენდაციებს. მონაცემთა სქემა შექმნილია ნორმალიზებული სტრუქტურით და ოპტიმიზირებულია შეკითხვის სიჩქარისთვის.

4. რეკომენდაციების მოდული – ეფუძნება ფიქსირებულ წესებზე აგებულ ალგორითმს, რომელიც აანალიზებს ხარჯვით ქცევას და მომხმარებლის მიზნებთან მიმართებით სთავაზობს ინდივიდუალურ რჩევებს. მომავლისთვის გათვალისწინებულია მანქანური სწავლის ალგორითმის ინტეგრაცია.

**მონაცემთა მიმოცვლის დიაგრამა:**  
მომხმარებელი ინტერფეისიდან აგზავნის მოთხოვნას → სერვერი ამოწმებს ავტორიზაციას (JWT) → ამუშავებს მონაცემს → უკავშირდება ბაზას და აბრუნებს შესაბამის ინფორმაციას ან რეკომენდაციას → პასუხი ბრუნდება მომხმარებლის ეკრანზე.

ეს არქიტექტურა ქმნის მოქნილ, გაფართოებად და უსაფრთხო სისტემას, რომელიც მიზნად ისახავს პერსონალიზებული ფინანსური მენეჯმენტის რეალურ მხარდაჭერას.

# პროგრამული სისტემის ფუნქციონირების აღწერა

FrontEnd:

საჭიროა ლინუქსის ოპერაციული სისტემა, მაგალითად: ubuntu 22.04

ასევე ios სმარტფონზე დააყენეთ აპლიკაცია ExpoGo

1. დაკლონეთ კოდის რეპოზიტორია: git pull (link)
2. დააყენეთ npm პაკეტი: apt install npm
3. დააყენეთ expo-cli: npm install -g expo-cli
4. npx expo login
5. შექმენით .env ფაილი,რომელიც შეიცავს ბექენდის ენდფოინთს:

echo"EXPO\_PUBLIC\_API\_BASE\_URL=http://financer-backend.polandcentral.cloudapp.azure.com/" > .env

1. დასტარტეთ აპლიკაცია: npx expo start --tunnel --go
2. დაასკანერეთ QR კოდი რომელსაც გამოიტანს კონსოლში და გაგიხსნით აპლიკაციაში: Expo Go

BackEnd:

საჭიროა ლინუქსის ოპერაციული სისტემა, მაგალითად: ubuntu 22.04

1. დაკლონეთ კოდის რეპოზიტორია:

git pull (link)

2. დააყენეთ საჭირო პაკეტები:

docker.io და docker-compose

3. ჩადექით დაკლონილ პროექტში და დასტარტეთ აპლიკაცია:

docker-compose --env-file .env up -d

# დასკვნა

პროექტის ფარგლებში შეიქმნა ინოვაციური მობილური აპლიკაციის კონცეფცია და ტექნიკური საფუძველი, რომელიც მომხმარებლებს საშუალებას აძლევს ეფექტურად მართონ საკუთარი ფინანსები. საწყის ამოცანად განისაზღვრა ისეთი პლატფორმის შექმნა, რომელიც არ შემოიფარგლება მხოლოდ ხარჯების აღრიცხვითა და სტატისტიკური მონაცემებით, არამედ სთავაზობს მომხმარებლებს პერსონალიზებულ რჩევებს, ქცევაზე დაფუძნებულ ანალიზს და სტრატეგიულ დაგეგმარებას.

ჩვენმა გუნდმა წარმატებით შეიმუშავა აპლიკაციის ინტერფეისის დიზაინი UI/UX პრინციპების გათვალისწინებით, დაგეგმა და დანერგა მონაცემთა ბაზა, შემუშავდა ალგორითმები პერსონალიზებული რეკომენდაციებისათვის, ასევე განხორციელდა აპლიკაციის ძირითადი ფუნქციონალის ტესტირება. შედეგად მივიღეთ პროდუქტისთვის ტექნიკურად მზა საფუძველი, რომელიც სრულად პასუხობს დასახულ მიზნებს.

აპლიკაციას აქვს დიდი განვითარების პოტენციალი. მომავალ ეტაპებზე შესაძლებელია მანქანური სწავლის ალგორითმების ინტეგრაცია, რაც რეკომენდაციების ხარისხს კიდევ უფრო გააუმჯობესებს. აგრეთვე, შესაძლებელია სხვადასხვა ფინანსურ ინსტიტუტებთან — ბანკებთან, გადახდის სისტემებთან და სხვა სერვისებთან — პირდაპირი ინტეგრაციის განხორციელება, რაც მომხმარებლისთვის აპლიკაციას კიდევ უფრო მოქნილსა და მრავალფუნქციურს გახდის.

მოკლედ, პროექტის შედეგი სრულად შეესაბამება საწყის ამოცანას და ქმნის მყარ საფუძველს იმისთვის, რომ პროდუქტი რეალურ, ფართო მომხმარებელზე გათვლილ, მაღალი ღირებულების მქონე ფინანსურ ინსტრუმენტად იქცეს.

# ლიტერატურა

1. Palantir - Why Traditional Approaches to Continuous Deployment Don’t Work Today

<https://blog.palantir.com/why-traditional-approaches-to-continuous-deployment-dont-work-today-b5a6c33cc754>

1. Continuous Deployments != Continuous Disruptions

<https://www.wagslane.dev/posts/continuous-deployments-arent-continuous-disruptions/>

1. Microsoft - Recommendations for standardizing tools and processes

<https://learn.microsoft.com/en-us/azure/well-architected/operational-excellence/tools-processes>

1. Testing of financial app

<https://www.cigniti.com/blog/factors-to-consider-while-testing-financial-apps/?utm_source=chatgpt.com>

1. Test cases for financial app

<https://www.headspin.io/blog/test-cases-for-financial-app-testing?utm_source=chatgpt.com>

1. Testing types

<https://www.atlassian.com/continuous-delivery/software-testing/types-of-software-testing?utm_source=chatgpt.com>

1. Types of software testing

<https://www.geeksforgeeks.org/types-software-testing/>

1. React Native Documentation

<https://reactnative.dev/>

1. GraphQL

<https://graphql.org/>

1. *Stack Overflow*  
   <https://stackoverflow.com/>
2. *Reddit – r/reactnative*  
   <https://www.reddit.com/r/reactnative/new/>