|  |  |  |
| --- | --- | --- |
| VILNIAUS KOLEGIJA  ELEKTRONIKOS IR INFORMATIKOS FAKULTETAS  PROGRAMINĖS ĮRANGOS KATEDRA | | |
|  | | |
| "SHIELD PROTECTION" PROJEKTO APRAŠAS | | |
| HIBRIDINĖS MOBILIOSIOS APLIKACIJOS  KD - 6531BX028 PI17B | | |
| STUDENTAI | (PARAŠAS)  2020-01-02 | MINDAUGAS BURVYS |
| DĖSTYTOJA | (PARAŠAS)  2020-01-02 | J. ZAILSKAS |
|  |  |  |
|  |  |  |
| 2018 | | |

# Turinys

[Turinys 2](#_Toc28848996)

[Įvadas 3](#_Toc28848997)

[1. Užduoties formulavimas 4](#_Toc28848998)

[Funkciniai reikalavimai 4](#_Toc28848999)

[Nefunkciniai reikalavimai 4](#_Toc28849000)

[1. Užduoties analizė 5](#_Toc28849001)

[2. Programinės realizacijos aprašymas 7](#_Toc28849002)

[3. Naudotojo Instrukcija 14](#_Toc28849003)

[Išvados ir pasiūlymai 15](#_Toc28849004)

[Informacijos šaltiniai 16](#_Toc28849005)

# Įvadas

Šio projekto yra skirtas pademonstruoti su React Native framework‘u įgytas per semestrą žinias ir sukurti projektą, savo nuožiūra pasirinkta tema, kuris atliktų konkretų darbą, pasinaudojant mūsų įgytomis žiniomis.

**Užduoties tikslas**:

Sukurti sistemą React native kalba, kurioje būtų įgyvendintos šios užduotys:

1. Sukurta mobilioji programėlė (panaudoti API’s, tai yra jutikliai).
2. Sukurta interneto svetainė (ar kita mobilioji programėlė), kurios bendrauja tarpusavyje.
3. Prijungta duomenų bazė (gali būti bet kokia SQL ar NoSQL žymėjimo kalbų duomenų bazė).
4. Pateikta sistemos dokumentacija. Programinis kodas saugomas GitHub ar GitLab sistemose.
5. Bendravimas tarp mobiliosios programėlės ir internetinės svetainės (ar mobilios aplikacijos) gali vykti per WebSocket’us arba WebService’us.

**Projekto tematika:**

Projekto tema yra sistema, valdanti saugomus objektus, bei informuojanti apie atitinkamų objektų saugojimo statusą. Taip pat ši sistema atvaizduoja sąrašą saugomų objektų su jų informacija (adresais, pavadinimais ir pan.), vartotojų (sargų) sąrašą, leidžia registruoti naujus sargus, bei objektus, taip pat valdyti sargų informaciją, pamatyti prisijungimo ir registravimosi laikus ir pasimėgauti katino nuotrauka.

**Projekto objektai:**

1. Valdymo aplikacija (React native aplikacija, sukurtas expo-cli pagalba).
2. Firebase duomenų bazė.
3. Vartotojų (objektų sargų) aplikacija (Android studio aplikacija).
4. Dokumentacija.

# Užduoties formulavimas

## Funkciniai reikalavimai

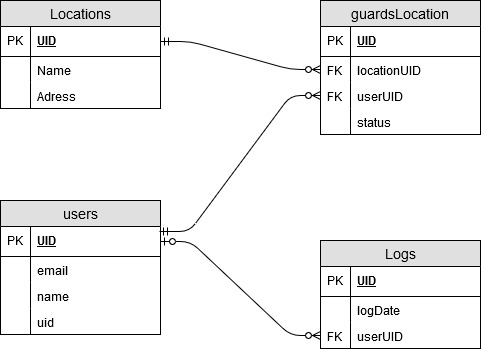
1. Sukurti autentifikavimo sistemą.
2. Sistema turi leisti registruoti naujus vartotojus.
3. Sukurti saugotinų objektų informacijos gavimą iš duomenų bazės ir duomenų atvaizdavimą.
4. Sukurti vartotojų informacijos gavimą iš duomenų bazės ir duomenų atvaizdavimą.
5. Sukurti prisijungimų informacijos (angl. logs) gavimą iš duomenų bazės ir duomenų atvaizdavimą.
6. Sukurti sargų saugomų objektų informacijos gavimą iš duomenų bazės ir duomenų atvaizdavimą.
7. Sistema turi leisti pridėti naujas saugomų objektų lokacijas.
8. Sukurtai sistemai turi būti sudaryta dokumentacija su aiškiai užduotyje išdėstytais jos punktais.

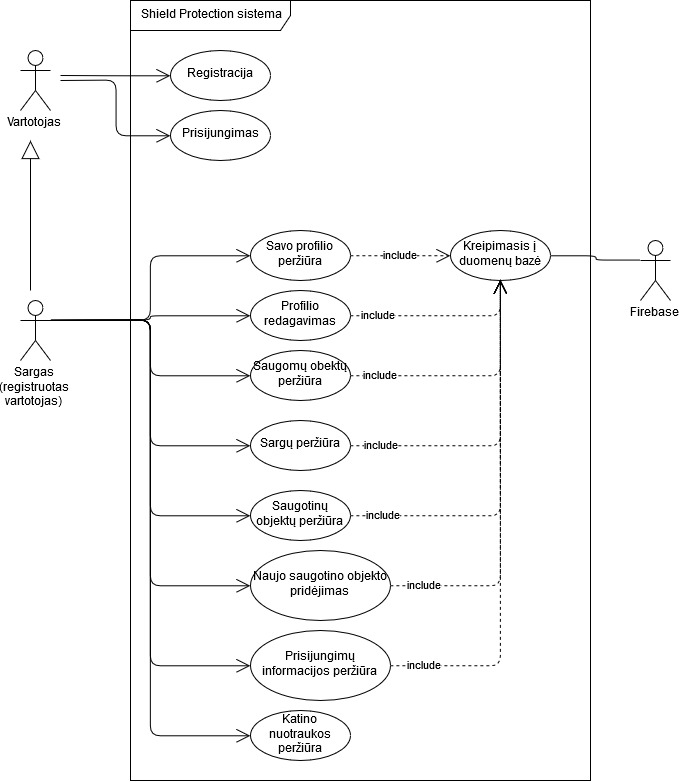
## Nefunkciniai reikalavimai

1. Sistema turi naudoti React Native framework’ą.
2. Sistema atvaizduoja informaciją pasinaudojant framework’o componentais ir jutikliais (angl. API’s).
3. Sistema naudoja Firebase duomenų bazę duomenų saugojimui ar vartotojų autentifikavimui.
4. Sistema turi būti funkciškai dali ir sistemos dalys (klasės ir metodai) turi atlikti tik jiems skirtą darbą (angl. single responsibility principle).
5. Sistema atvaizduoja informaciją aiškiais ir nuspėjamais vartotojui veiksmais.

# Užduoties analizė

Firebase DB ER ToDo!!!!!!





Sistemos use case diagrama ToDo!!!!!!!!!!!!

# Programinės realizacijos aprašymas

**Sistemos programinės realizacijos failų sąrašas:**

* actions
  + user.js
* config
  + Firebase.js
* images
  + ImagesExample.js
  + logo.png
  + LogoView.js
  + Zuu\_the\_sleepy\_cat.jpg
* navigation
  + SwitchNavigator.js
* reducers
  + index.js
* screens
  + EditProfile.js
  + GuardInfo.js
  + Locations.js
  + LocationsAdd.js
  + Login.js
  + Logs.js
  + Profile.js
  + ProtectedLocations.js
  + Signup.js
  + SleepyCat.js
* .env
* App.js

**Failų sandara ir veikimas:**

* **user.js**

Šis failas turi funkcijas ir veiksmų konstantas, skirtas prisijungti ir užregistruoti vartotojus. Šiam darbui atlikti yra naudojamas importuoto Firebase.js failo konstantos skirtos susisiekti su duomenų baze.

* **Failo sandara:**
  + updateEmail(email) funkcija
  + updatePassword(password) funkcija
  + login() funkcija
  + getUser(uid) funkcija
  + signup() funkcija
  + UPDATE\_EMAIL konstanta
  + UPDATE\_PASSWORD konstanta
  + LOGIN konstanta
  + SIGNUP konstanta
* updateEmail(email) funkcija atnaujina vartotojo objekto el. pašto adresą, kuris yra gautas iš duomenų bazės.
* updatePassword(password) funkcija atnaujina vartotojo objekto slaptažodį, kuris yra gautas iš duomenų bazės.
* login() funkcija gauna lokalaus vartotojo objekto duomenis (el. paštą ir slaptažodį), jais bando prisijunti, kreipiantis į Firebase autentifikavimo sistemą, sukuria prisijungimo statuso pranešimą vartotojui (ar prisijungimas sėkmingas ar ne) ir registruoja prisijungimo laiką duomenų bazėje.
* getUser(uid) funkcija kreipiasi į Firebase duomenų bazę ir pagal turimą vartotojo UID bando gauti atitinkamo vartotojo duomenis iš Firebase duomenų bazės.
* signup() funkcija gauna lokalaus vartotojo objekto duomenis (el. paštą ir slaptažodį), jais bando registruotis, kreipiantis į Firebase autentifikavimo sistemą ir jei šis veiksmas buvo sėkmingas, išsaugo vartotojo profilio duomenis duomenų bazėje, ir registruoja prisijungimo laiką duomenų bazėje.
* **Firebase.js**

Šis failas skirtas susieti sistemą su Firebase duomenų baze, tam naudojant Firebase ir „firebase/firestore“ modulius, bei atitinkamus duomenis, išsaugotus projekto aplinkos faile (.env), būtinus susieti Firebase su atitinkama duomenų baze.

* **Failo sandara:**
  + firebaseConfig konstanta, išsauganti būtinus prisijungimui duomenis, kaip objektą.
  + db konstanta, kuri skirt sutrumpini kodo užklausų, kurios kreipsis į duomenų bazę rašymui.
* **logo.png ir Zuu\_the\_sleepy\_Cat.png**

Tai paveikslėlių failai, kurie yra naudojami sistemoje.

* **ImagesExample.js**

Tai failas turintis metodą, kuris atvaizduoja Zuu\_the\_sleepy\_Cat.png paveikslėlį, su atitinkamais parametrais.

* **Failo sandara:**
  + ImagesExample() funkcija, atvaizduojanti HTML View elementą, kuris turi HTML Image elementą, su atitinkamu paveikslėliu ir jo stiliumi (dydis ir pan.).
* **LogoView.js**

Tai failas turintis metodą, kuris atvaizduoja logo.png paveikslėlį, su atitinkamais parametrais.

* **Failo sandara:**
  + LogoView() funkcija, atvaizduojanti HTML View elementą, kuris turi HTML Image elementą, su atitinkamu paveikslėliu ir jo stiliumi (dydis ir pan.).
* **SwitchNavigator.js**

Tai failas skirtas sukurti SwitchNavigator konstantą, skirta greičiau ir paprasčiau valdyti perėjimą tarp atskirų sistemos langų. Tam yra naudojamas react-navigation modulis, kuris leidžia sukurti šį objektą, kuris turi sąrašą langų, į kuriuos galima pereiti, taip pat ir saugodamas pagrindinį (pirmą šio navigatoriaus atidaromą) langą.

* **Failo sandara:**
  + SwitchNavigator konstanta.
* **index.js**

Tai failas skirtas sukurti user ir rootReducer konstantas. Šios konstantos yra naudojamos, kaip tarpinis elementas, tarp duomenų bazės ir sistemos, apibrėžiant tik tam tikrus veiksmus, kuriuos sistema gali atlikti, pasinaudojant redux moduliu.

* **Failo sandara:**
  + user konstanta, apibrėžianti prisijungimo, registracijos, el. pašto keitimo ir slaptažodžio keitimo veiksmus.
  + rootReducer konstanta, susiejanti visus joje paminėtų veiksmų konstantų objektus.
* **EditProfile.js**

Failas skirtas profilio redagavimo klasei ir jos atvaizdavimui (render).

* **Failo sandara:**
  + EditProfile klasė
    - handleBack() metodas skirtas vykdyti, grįžimo atgal į ankstesnį langą, funkciją.
    - render() metodas atvaizduojantis jame aprašyto HTML žymėjimo kalbos su įterptomis Java script kodo detalėmis vaizdą.
  + styles kosntanta, skirta išsaugoti stiliaus (StyleSheer) objekto duomenis, HTML žymėjimo kalbos kodui sutrumpinti.
* **GuardInfo.js**

Failas skirtas sargų (registruotų vartotojų) informacijos atvaizdavimui (render).

* **Failo sandara:**
  + GuardInfo klasė
    - users state’as, skirtas saugoti iš duomenų bazės gautų duomenų objektus (vartotojų vardus ir el. paštus)
    - handleBack() metodas skirtas vykdyti, grįžimo atgal į ankstesnį langą, funkciją.
    - renderItem(obj) metodas skirtas atvaizduoti vieno objekto (išsaugoto state’e vartotojo vieneto) duomenis.
    - componentDidMount() metodas pradedamas vykdyti prieš atliekant kitus klasės veiksmus, skirtas gauti informacijai, kuri bus atvaizduota render() metode, iš duomenų bazės.
    - render() metodas atvaizduojantis jame aprašyto HTML žymėjimo kalbos su įterptomis Java script kodo detalėmis vaizdą.
  + styles kosntanta, skirta išsaugoti stiliaus (StyleSheer) objekto duomenis, HTML žymėjimo kalbos kodui sutrumpinti.
* **Locations.js**

Failas skirtas saugotinų objektų (lokacijų) informacijos atvaizdavimui (render).

* **Failo sandara:**
  + Locations klasė
    - locations state’as, skirtas saugoti iš duomenų bazės gautų duomenų objektus (saugotino objekto ID, pavadinimą ir adresą)
    - handleBack() metodas skirtas vykdyti, grįžimo atgal į ankstesnį langą, funkciją.
    - handleAdd() metodas skirtas vykdyti, perėjimo į lokacijos pridėjimo lango, funkciją.
    - renderItem(obj) metodas skirtas atvaizduoti vieno objekto (išsaugoto state’e lokacijos vieneto) duomenis.
    - componentDidMount() metodas pradedamas vykdyti prieš atliekant kitus klasės veiksmus, skirtas gauti informacijai, kuri bus atvaizduota render() metode, iš duomenų bazės.
    - render() metodas atvaizduojantis jame aprašyto HTML žymėjimo kalbos su įterptomis Java script kodo detalėmis vaizdą.
  + styles kosntanta, skirta išsaugoti stiliaus (StyleSheer) objekto duomenis, HTML žymėjimo kalbos kodui sutrumpinti.
* **LocationsAdd.js**

Failas skirtas papildomų saugotinų objektų (lokacijų) informacijos pridėjimui į duomenų bazę.

* **Failo sandara:**
  + LocationsAdd klasė
    - locName state’as, skirtas saugoti saugotino objekto pavadinimą.
    - locAdress state‘as skirtas saugoti saugotino objekto adresą.
    - handleBack() metodas skirtas vykdyti, grįžimo atgal į ankstesnį langą, funkciją.
    - handleAdd() metodas skirtas vykdyti lokacijos pridėjimo funkciją ir grįžimą į lokacijų langą.
    - render() metodas atvaizduojantis jame aprašyto HTML žymėjimo kalbos su įterptomis Java script kodo detalėmis vaizdą.
  + styles kosntanta, skirta išsaugoti stiliaus (StyleSheer) objekto duomenis, HTML žymėjimo kalbos kodui sutrumpinti.
* **Login.js**

Failas skirtas sargų (registruotų vartotojų) informacijos atvaizdavimui (render). Ši klasė naudojasi Redux tarpiniu elementu, kad perimtų vartotojo duomenis iš duomenų bazės.

* **Failo sandara:**
  + Login klasė
    - handleLogin() metodas skirtas vykdyti, vartotojo prisijungimo funkciją, kviečiant user.js faile aprašytą veiksmą.
    - componentDidMount() metodas pradedamas vykdyti prieš atliekant kitus klasės veiksmus, skirtas gauti vartotojo informacijai iš duomenų bazės, jeigu yra prisijungęs vartotojas ar gautas teisingas autentifikavimo pranešimo pasikeitimas.
    - render() metodas atvaizduojantis jame aprašyto HTML žymėjimo kalbos su įterptomis Java script kodo detalėmis.
  + styles kosntanta, skirta išsaugoti stiliaus (StyleSheer) objekto duomenis, HTML žymėjimo kalbos kodui sutrumpinti.
* **Logs.js**

Failas skirtas vartotojų prisijungimų laiko registravimui ir atvaizdavimui (render).

* **Failo sandara:**
  + Logs klasė
    - logs state’as, skirtas saugoti iš duomenų bazės gautų duomenų objektus (vartotojų prisijungimų laikus)
    - handleBack() metodas skirtas vykdyti, grįžimo atgal į ankstesnį langą, funkciją.
    - renderItem(obj) metodas skirtas atvaizduoti vieno objekto (išsaugoto state’e log‘o vieneto) duomenis.
    - componentDidMount() metodas pradedamas vykdyti prieš atliekant kitus klasės veiksmus, skirtas gauti informacijai, kuri bus atvaizduota render() metode, iš duomenų bazės.
    - render() metodas atvaizduojantis jame aprašyto HTML žymėjimo kalbos su įterptomis Java script kodo detalėmis.
  + styles kosntanta, skirta išsaugoti stiliaus (StyleSheer) objekto duomenis, HTML žymėjimo kalbos kodui sutrumpinti.
* **Profile.js**

Failas skirtas sargo (registruoto vartotojo) informacijos atvaizdavimui (render).

* **Failo sandara:**
  + Profile klasė
    - handleSignout() metodas skirtas vykdyti atsijungimo ir grįžimo į prisijungimo langą funkcijai.
    - handleEdit() metodas skirtas pereiti į profilio redagavimo puslapį.
    - handleLocations() metodas skirtas pereiti į lokacijų atvaizdavimo puslapį.
    - handleCat() ypatingai būtinas metodas skirtas pereiti į SleepyCat atvaizdavimo puslapį.
    - handleProtected() metodas skirtas pereiti į saugomų objektų informacijos pusalapį
    - handleGuards() metodas skirtas pereiti į sargų informacijos atvaizdavimo puslapį.
    - handleLogs() metodas skirtas pereiti į vartotojų prisijungimų laikų atvaizdavimo puslapį.
    - render() metodas atvaizduojantis jame aprašyto HTML žymėjimo kalbos su įterptomis Java script kodo detalėmis.
  + styles kosntanta, skirta išsaugoti stiliaus (StyleSheer) objekto duomenis, HTML žymėjimo kalbos kodui sutrumpinti.
* **ProtectedLocations.js**

Failas skirtas saugomų objektų informacijos atvaizdavimui (render).

* **Failo sandara:**
  + ProtectedLocations klasė
    - GuardsLocation state’as, skirtas saugoti iš duomenų bazės gautų duomenų objektus saugomų objektų vietos ID, vartotojo ID, objekto saugojimo statusą, lokacijos pavadinimą ir lokacijos adresą).
    - handleBack() metodas skirtas vykdyti, grįžimo atgal į ankstesnį langą, funkciją.
    - renderItem(obj) metodas skirtas atvaizduoti vieno objekto (išsaugoto state’e saugomo objketo vieneto) duomenis.
    - componentDidMount() metodas pradedamas vykdyti prieš atliekant kitus klasės veiksmus, skirtas gauti informacijai, kuri bus atvaizduota render() metode, iš duomenų bazės.
    - render() metodas atvaizduojantis jame aprašyto HTML žymėjimo kalbos su įterptomis Java script kodo detalėmis.
  + styles kosntanta, skirta išsaugoti stiliaus (StyleSheer) objekto duomenis, HTML žymėjimo kalbos kodui sutrumpinti.
* **Signup.js**

Failas skirtas vartotojo registracijai ir registracijos vaizdo atvaizdavimui.

* **Failo sandara:**
  + Signup klasė
    - handleSignUp() metodas skirtas vykdyti registracijos funkciją, naudojantis user.js faile aprašytais veiksmais.
    - render() metodas atvaizduojantis jame aprašyto HTML žymėjimo kalbos su įterptomis Java script kodo detalėmis.
  + styles kosntanta, skirta išsaugoti stiliaus (StyleSheer) objekto duomenis, HTML žymėjimo kalbos kodui sutrumpinti.
* **SleepyCat.js**

Ypatingai būtinas failas skirtas atvaizdavimui katino paveikslėlį.

* **Failo sandara:**
  + SleepyCat klasė
    - handleBack() metodas skirtas vykdyti, grįžimo atgal į ankstesnį langą, funkciją.
    - render() metodas atvaizduojantis jame aprašyto HTML žymėjimo kalbos su įterptomis Java script kodo detalėmis.
  + styles kosntanta, skirta išsaugoti stiliaus (StyleSheer) objekto duomenis, HTML žymėjimo kalbos kodui sutrumpinti.
* **.env**

Failas skirtas išsaugoti Firebase prisijungimui būtinus duomenis.

* **Failo sandara:**
  + Tekstinio tipo duomenys, nusakantys Firebase prisijungimo parametrus.
* **App.js**

Pradinis aplikacijos failas, kuriuo paleidžiama programa. Jame tik susiejamas redux tarpininkas su programa ir paleidžiamas SwitchNavigator navigatorius.

* Failo sandara:
  + App klasė, kuri yra pradinė (main) aplikacijos klasė
    - render() metodas skirtas atvaizduoti pradiniam vaizdui, kuris plėtojamas tik kitose klasėse.

# Naudotojo Instrukcija

# Išvados ir pasiūlymai

Šio semestro metu pramokome rašyti React Native framework‘o internetines, bei mobiliąsias aplikacijas. Kuriant ši projektą buvo stengtasi pademonstruoti šiuos sugebėjimus, bei HTML ir Java script kalbų mąstymą, kuriant savo projektą.

Norint toliau plėsti šią sistemą reiktų atsižvelgti į didesnį kiekį duomenų, kurie praverstų tokios sistemos darbui atitinkamose terpėse. Taip pat norint konkrečiau stebėti ir valdyti tam tikrus duomenis, praverstų geresnis duomenų bazės dizainas ir valdymas perkeltas į atitinkamus reducer‘ius ir veiksmus. Taip pat tobulinant šią sistemą padėtų didesnės apimties informacijos tyrimas, sąsajos, informacijos atvaizdavimo tipams, informacijos rinkimui ir informacijos rašymui į sistemą ir sistemos suvokimui ir patogesniam išdėstymui planuoti, ką galima būtų nesunku padaryti turint didesnį laiko tarpą sistemos planavimui. Norint geriau valdyti HTML turinio atvaizdavimą, taip pat praverstų didesnis HTML ir JS supratimas, bent baziniu lygiu, o ne nuliniu, kokiu dalis studentų pradėjo šį semestrą.

# Informacijos šaltiniai

1. UML diagramų standartų aprašymas. <https://www.uml-diagrams.org/>
2. Firebase konsolė ir dokumentacija. <https://console.firebase.google.com/>
3. React native dokumentacija. <https://facebook.github.io/react-native/docs>
4. React Redux. <https://react-redux.js.org/>
5. React native – React Redux Firebase. <https://react-redux-firebase.com/docs/integrations/react-native.html>