



# **Państwowa Wyższa Szkoła Zawodowa w Tarnowie**

Dokumentacja – Aplikacja treningowa  
Podstawy programowania systemów mobilnych II  
Mgr inż. Adam Pieprzycki

Autorzy:  
Achwat Mariusz  
Polek Leszek  
Ryba Marek

# Spis treści

1. Wstęp.....	3
2. Sposób instalacji aplikacji.....	3
3. Fragmenty kodu aplikacji.....	4
4. Proponowane dalsze możliwości rozbudowy.....	7
5. Instrukcja obsługi .....	7
6. Uruchomienie serwera .....	7

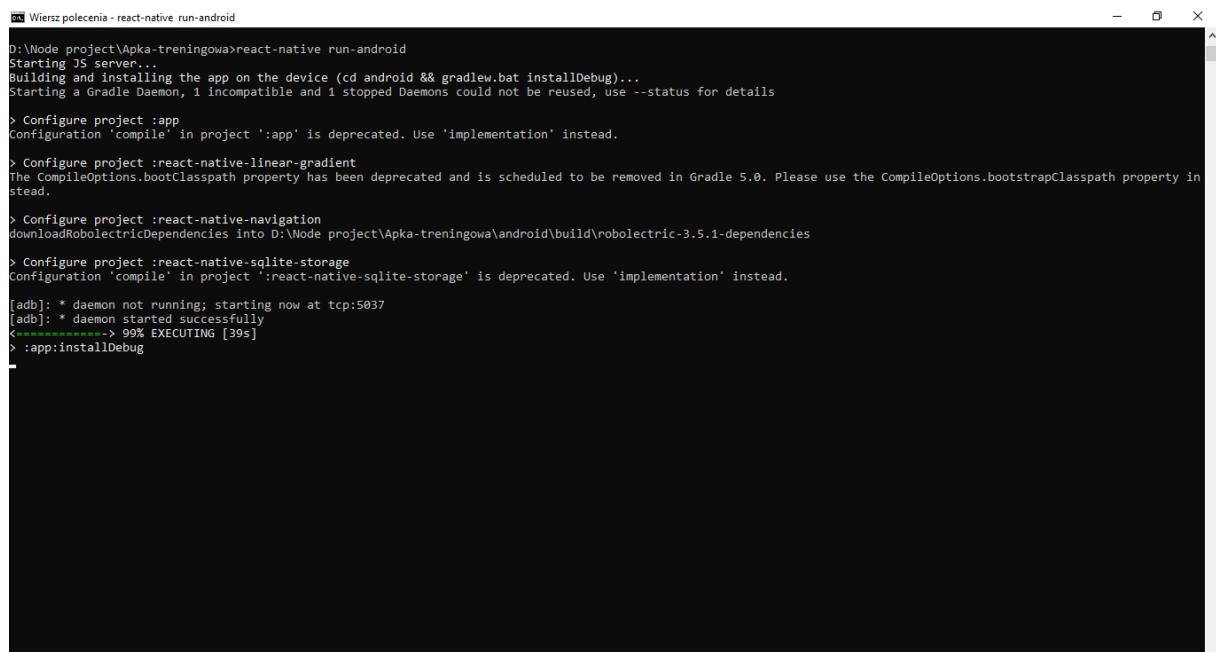
# 1. Wstęp

Celem projektu było stworzenie aplikacji treningowej, która pozwoli użytkownikowi zadbać o swoją kondycję fizyczną oferując mu różne typy ćwiczeń. Dodatkowo aplikacja będzie powiadamiała użytkownika o treningu. Projekt został stworzony na potrzeby przedmiotu Podstawy programowania systemów mobilnych II pod przewodnictwem mgr inż. Adama Pieprzyckiego. Do wykonania projektu użyto następujących narzędzi:

- IntelliJ Idea
- Android Studio
- Język JavaScript
- Framework React Native
- DB Browser

## 2. Sposób instalacji aplikacji

Na chwilę obecną aplikację można zainstalować przy użyciu wiersza poleceń na komputerze poprzez wpisanie komendy: `react-native run-android`.



```
Wiersz polecenia - react-native run-android
D:\Node project\Apka-treningowa>react-native run-android
Starting JS server...
Building and installing the app on the device (cd android && gradlew.bat installDebug)...
Starting a Gradle Daemon, 1 incompatible and 1 stopped Daemons could not be reused, use --status for details

> Configure project :app
Configuration 'compile' in project ':app' is deprecated. Use 'implementation' instead.

> Configure project :react-native-linear-gradient
The CompileOptions.bootstrapClasspath property has been deprecated and is scheduled to be removed in Gradle 5.0. Please use the CompileOptions.bootstrapClasspath property instead.

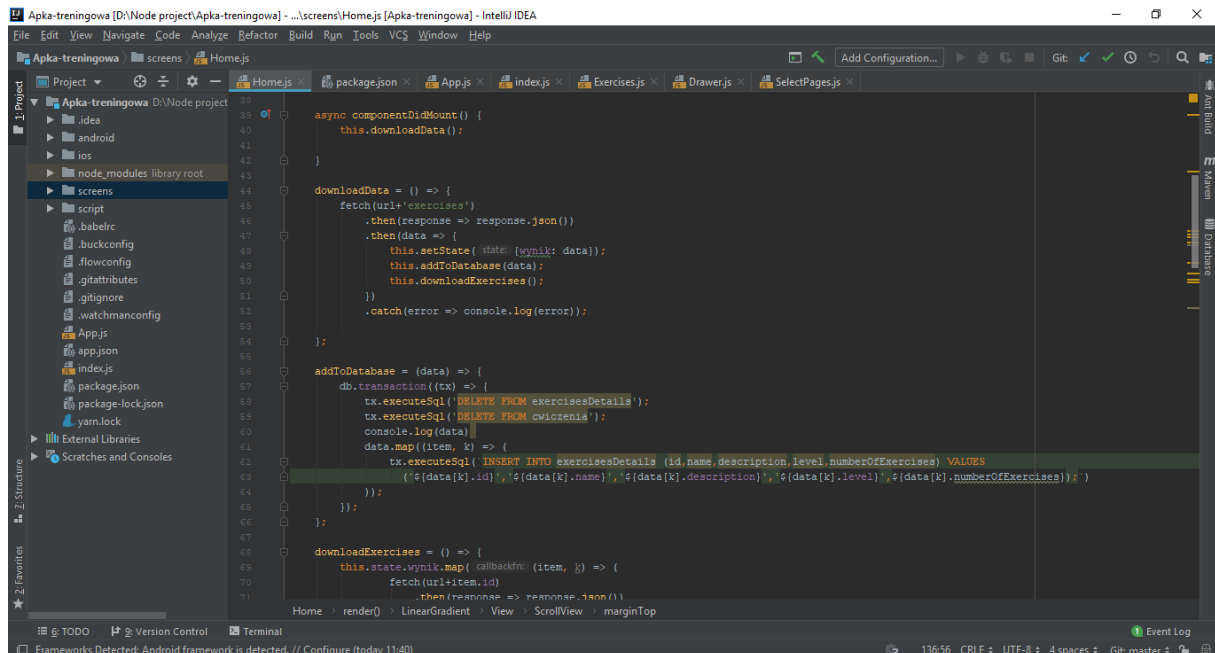
> Configure project :react-native-navigation
downloadRobolectricDependencies into D:\Node project\Apka-treningowa\android\build\robolectric-3.5.1-dependencies

> Configure project :react-native-sqlite-storage
Configuration 'compile' in project ':react-native-sqlite-storage' is deprecated. Use 'implementation' instead.

[adb]: * daemon not running; starting now at tcp:5037
[adb]: * daemon started successfully
<-----> 99% EXECUTING [39s]
> :app:installDebug
```

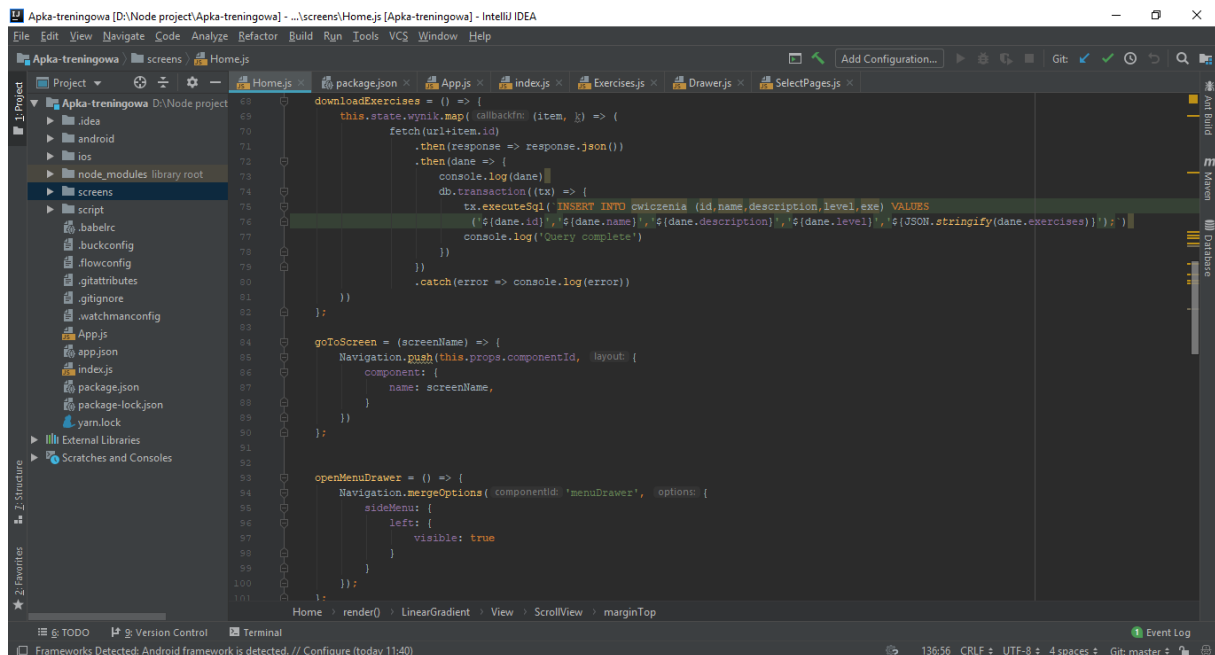
### 3. Fragmenty kodu aplikacji

Pobieranie i uzupełnianie bazy z ćwiczeniami:



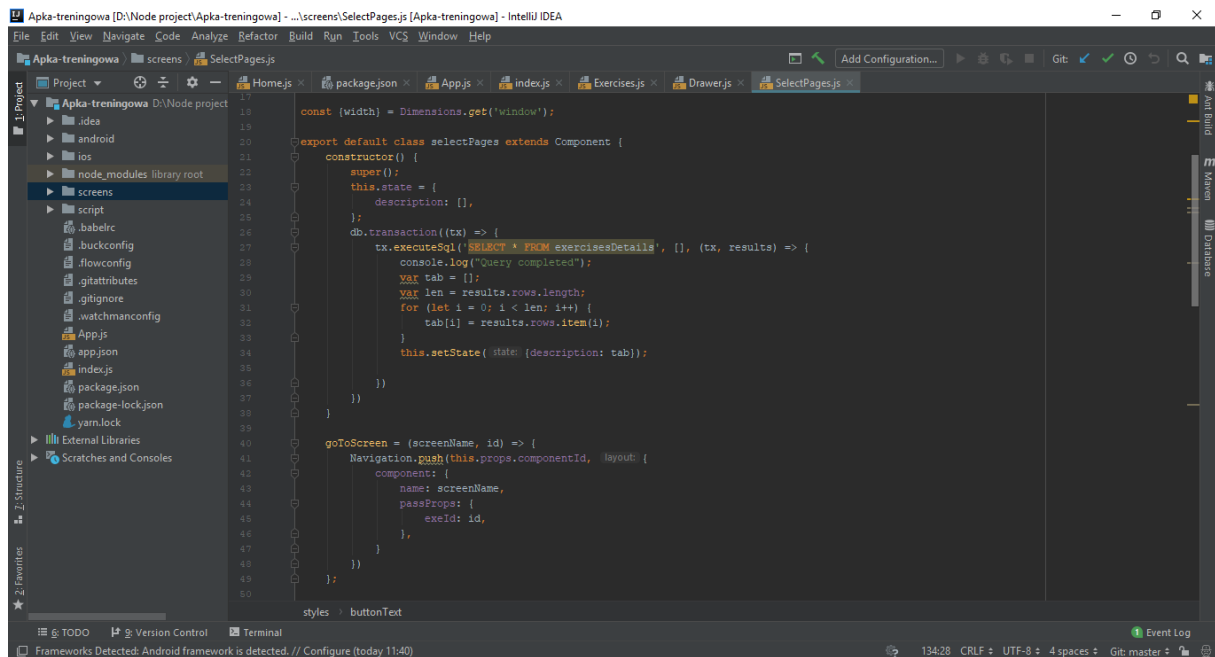
```
39 async componentDidMount() {
40   this.downloadData();
41 }
42
43
44 downloadData = () => {
45   fetch(url+'exercises')
46     .then(response => response.json())
47     .then(data => {
48       this.setState({wynik: data});
49       this.addToDatabase(data);
50       this.downloadExercises();
51     })
52     .catch(error => console.log(error));
53 }
54
55
56 addDatabase = (data) => {
57   db.transaction((tx) => {
58     tx.executeSql('DELETE FROM exercisesDetails');
59     tx.executeSql('DELETE FROM cwiczenia');
60     console.log(data)
61     data.map((item, k) => {
62       tx.executeSql('INSERT INTO exercisesDetails (id,name,description,level,numberOfExercises) VALUES
63         ('+data[k].id+', '+data[k].name+', '+data[k].description+', '+data[k].level+', '+data[k].numberOfExercises+')');
64     });
65   });
66 }
67
68
69 downloadExercises = () => {
70   this.state.wynik.map((callback) => {
71     fetch(url+item.id)
72       .then(response => response.json())
73       .then(data => {
74         console.log(data)
75         db.transaction((tx) => {
76           tx.executeSql('INSERT INTO cwiczenia (id,name,description,level,exe) VALUES
77             ('+data.id+', '+data.name+', '+data.description+', '+data.level+', '+JSON.stringify(data.exercises)')');
78           console.log('Query complete')
79         });
80       })
81       .catch(error => console.log(error))
82   });
83 }
84
85 goToScreen = (screenName) => {
86   Navigation.push(this.props.componentId, {
87     component: {
88       name: screenName,
89     }
90   });
91 }
92
93
94 openMenuDrawer = () => {
95   Navigation.mergeOptions(componentId: 'menuDrawer', options: {
96     sideMenu: {
97       left: {
98         visible: true
99       }
100     }
101   });
102 }
```

Przechodzenie po poszczególnych stronach aplikacji, otwieranie Drawer'a:

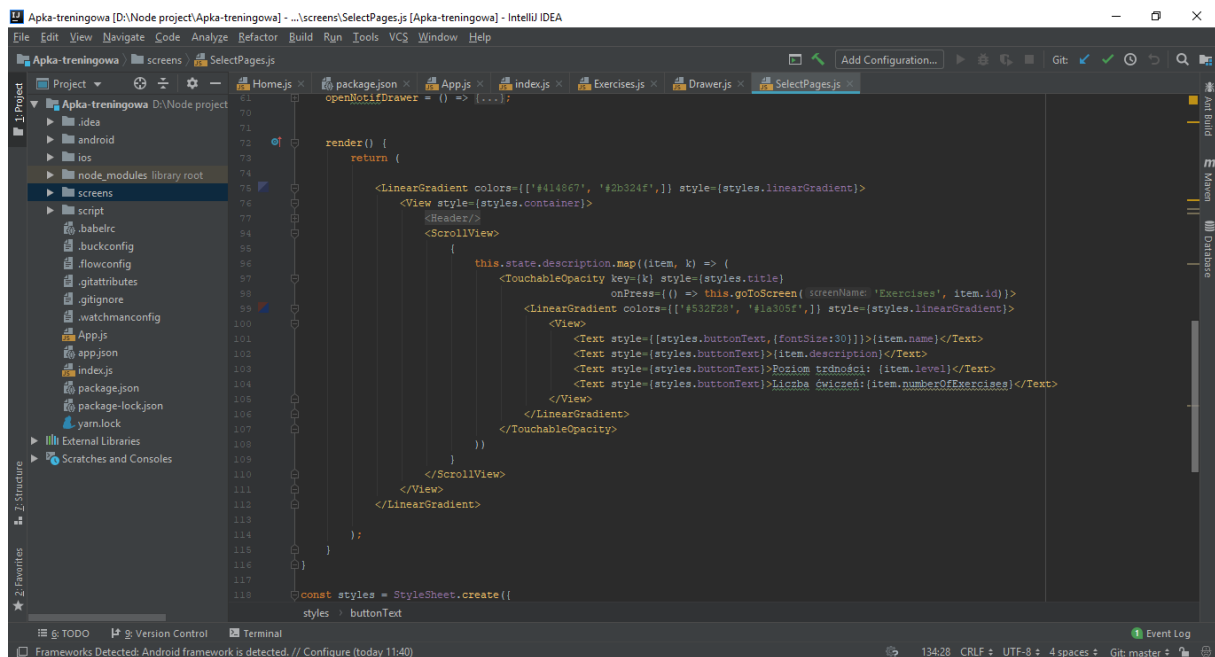


```
68 downloadExercises = () => {
69   this.state.wynik.map((callback) => {
70     fetch(url+item.id)
71       .then(response => response.json())
72       .then(data => {
73         console.log(data)
74         db.transaction((tx) => {
75           tx.executeSql('INSERT INTO cwiczenia (id,name,description,level,exe) VALUES
76             ('+data.id+', '+data.name+', '+data.description+', '+data.level+', '+JSON.stringify(data.exercises)')');
77           console.log('Query complete')
78         });
79       })
80       .catch(error => console.log(error))
81   });
82 }
83
84 goToScreen = (screenName) => {
85   Navigation.push(this.props.componentId, {
86     component: {
87       name: screenName,
88     }
89   });
90 }
91
92
93 openMenuDrawer = () => {
94   Navigation.mergeOptions(componentId: 'menuDrawer', options: {
95     sideMenu: {
96       left: {
97         visible: true
98       }
99     }
100   });
101 }
```

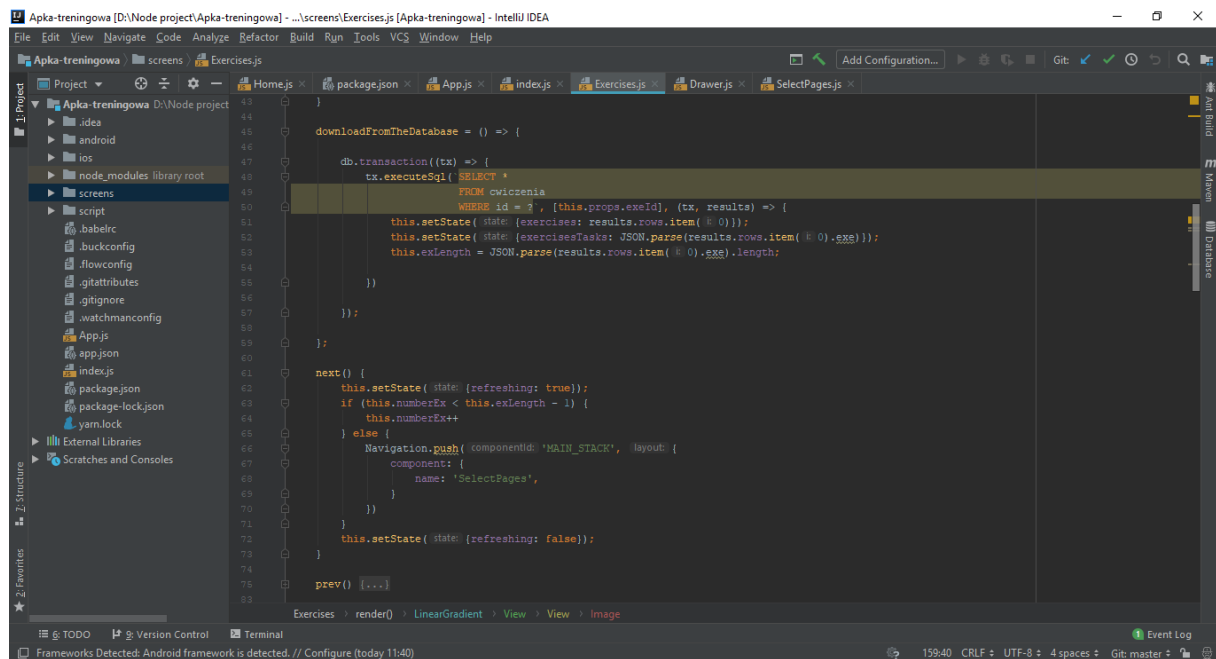
## Pobieranie opisów ćwiczeń:



## Generowanie przycisków wyboru ćwiczenia:

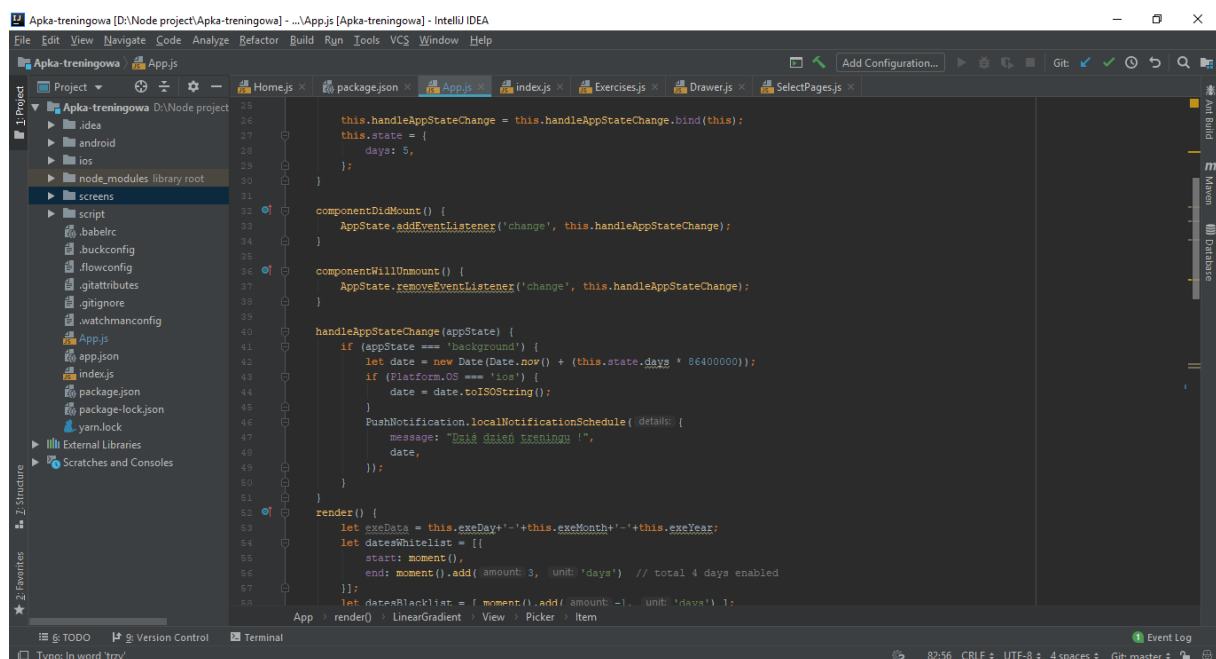


Pobieranie z bazy szczegółowych danych o ćwiczeniach, funkcja do przechodzenia po ćwiczeniach:



```
43 }
44
45 downloadFromTheDatabase = () => {
46
47   db.transaction((tx) => {
48     tx.executeSql( 'SELECT *
49     FROM cwiczenia
50     WHERE id = ? ', [this.props.exeId], (tx, results) => {
51       this.setState( {state: {exercises: results.rows.item( 0 )}});
52       this.setState( {state: {exercisesTasks: JSON.parse(results.rows.item( 0 ).exe)}});
53       this.exLength = JSON.parse(results.rows.item( 0 ).exe).length;
54     })
55   });
56
57   });
58
59 };
60
61 next() {
62   this.setState( {state: {refreshing: true}});
63   if (this.numberEx < this.exLength - 1) {
64     this.numberEx++
65   } else {
66     Navigation.push( {componentId: 'MAIN_STACK', layout: {
67       component: {
68         name: 'SelectPages',
69       }
70     })
71   }
72   this.setState( {state: {refreshing: false}});
73
74   prev() { ... }
75
76 }
```

Kod odpowiedzialny za notyfikacje:



```
25
26 this.handleAppStateChange = this.handleAppStateChange.bind(this);
27 this.state = {
28   days: 5,
29 };
30
31 componentDidMount() {
32   AppState.addEventListener('change', this.handleAppStateChange);
33 }
34
35 componentWillUnmount() {
36   AppState.removeEventListener('change', this.handleAppStateChange);
37 }
38
39 handleAppStateChange(appState) {
40   if (appState === 'background') {
41     let date = new Date(Date.now()) + (this.state.days * 86400000);
42     if (Platform.OS === 'ios') {
43       date = date.toISOString();
44     }
45     PushNotification.localNotificationSchedule( {details: {
46       message: "Twoja data treningowa !",
47       date,
48     }});
49   }
50 }
51
52 render() {
53   let exeDate = this.exeDay+"-"+this.exeMonth+"-"+this.exeYear;
54   let datesWhitelist = [
55     startDate: moment(),
56     endDate: moment().add( {amount: 3, unit: 'days'} ) // total 4 days enabled
57   ];
58   let datesBlacklist = [ moment().add( {amount: -1, unit: 'days'} ) ];
59 }
```

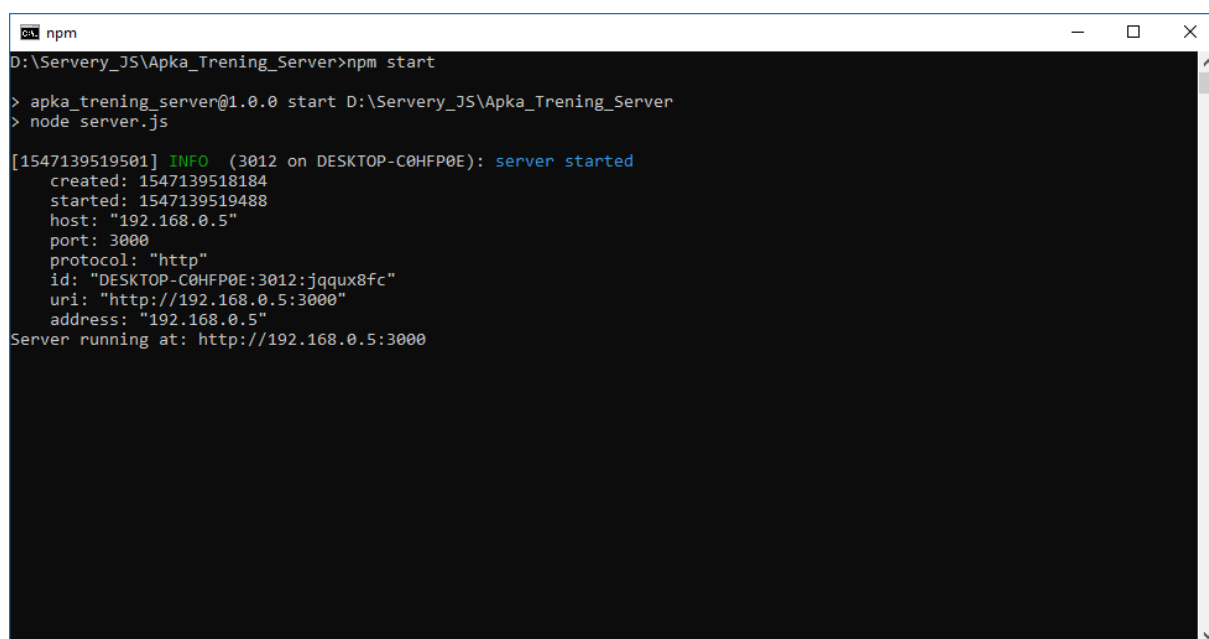
## 4. Proponowane dalsze możliwości rozbudowy

W dalszej rozbudowie, do aplikacji mogłaby zostać dodana historia wykonanych ćwiczeń, dzięki której użytkownik mógłby sprawdzać swoje postępy. Można wprowadzić system osiągnięć, który na swój sposób będzie mobilizował użytkownika do ćwiczeń. W przyszłości można udostępnić aplikację w sklepie Google Play, aby więcej użytkowników mogło z niej skorzystać. Można dodać również funkcję notatnika.

## 5. Instrukcja obsługi

Po uruchomieniu aplikacji wyświetla się menu główne, w którym do wyboru mamy różne treningi. Po przeciągnięciu palcem po ekranie od lewej do prawej strony wyświetli nam się menu pomocnicze, z którego też będzie można wybrać odpowiedni trening. Po przeciągnięciu palcem po ekranie od prawej do lewej strony wyświetli się kalendarz. Po wybraniu treningu na ekranie wyświetli się nazwa ćwiczenia, krótki opis jak je wykonać oraz czas jaki jest przewidziany na jego wykonanie. Pojawią się też trzy przyciski: Start – rozpoczyna odliczanie czasu ćwiczenia, Next – przechodzi do następnego ćwiczenia, Prev – przechodzi do poprzedniego ćwiczenia. Po wykonaniu wszystkich ćwiczeń z danego treningu i wciśnięciu przycisku Next, aplikacja wraca do ekranu z wyborem treningu. Na stronie głównej znajduje się przycisk Info, po kliknięciu którego wyświetla się alert z wersją aplikacji oraz jej autorami.

## 6. Uruchomienie serwera



```
npm
D:\Sery_J5\Apka_Trening_Server>npm start

> apka_trening_server@1.0.0 start D:\Sery_J5\Apka_Trening_Server
> node server.js

[1547139519501] INFO (3012 on DESKTOP-C0HFP0E): server started
  created: 1547139518184
  started: 1547139519488
  host: "192.168.0.5"
  port: 3000
  protocol: "http"
  id: "DESKTOP-C0HFP0E:3012:jqqux8fc"
  uri: "http://192.168.0.5:3000"
  address: "192.168.0.5"
Server running at: http://192.168.0.5:3000
```