

**Kucheriava O.M.**

Ph.D. (Physics and Maths),  
Associate Professor,  
*Department of Computerized  
Control Systems  
Kyiv, Ukraine*

**Holeho N.M.**

Senior lecturer at the  
Department of Computerized  
Control Systems,  
*National Aviation University  
Kyiv, Ukraine*

**Bachynska L.L.**

Senior lecturer at the  
English Department,  
*National University of  
Kyiv-Mohyla Academy  
Kyiv, Ukraine*

## **SOFTWARE MODULE OF CREATING CROSSPLATFORM APPLICATIONS BASED ON REACT/REACT NATIVE TECHNOLOGIES**

Mobile platforms are becoming more and more popular, so modern software has to be supported not only by stationary computers but also by mobile devices, such as tablet computers, smartphones and others. But not all the mechanisms of creating crossplatform applications are optimal, as there are not only many devices but also many various platforms on which these devices work. That is why the problem of finding flexible solutions in the area of crossplatform software applications still exists. The most important recent achievement is creating of crossplatform and untyped programming language JavaScript, due to which it has

become possible to create universal frameworks for interaction between platforms.

Software module of creating crossplatform applications for convenient and rapid work out process has been developed. Work with the module involves creating of a web application and mobile application which is developed on the basis of two-level client-server architecture and with frameworks React and React Native which enable the application to work not only on mobile devices but also on any devices that have software for browsing web pages.

React Native is a mobile framework which enables creating of applications using only JavaScript language. Yet differently from other hybrid mobile technologies, mobile web application is not being created. JavaScript database code compiles to the mobile application which does not differ from iOS application that is built using Objective-C or Android application using Java language. It means that React Native technology gives advantages for both native and hybrid mobile applications. Currently there are enough frameworks which use JavaScript for creating of iOS and Android applications [1].

The developed module of creating of crossplatform applications implements the main functionality for application creation and adapting to any devices and a platform. Owing to the module it is possible to provide support after creating of crossplatform applications, too.

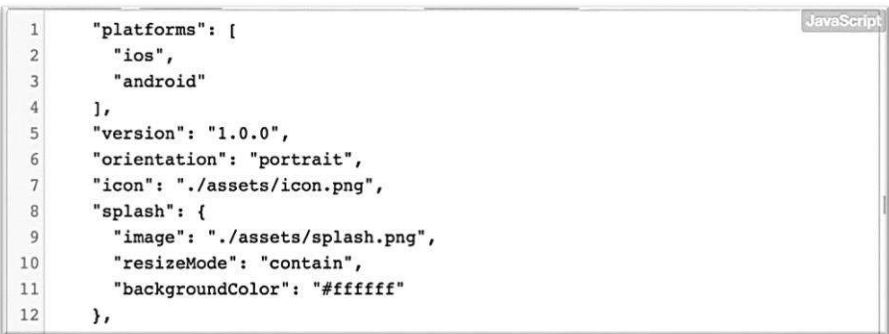
Crossplatform application which is created with the module contains such functionality: different platform support, authentication module, obtaining data from a server, module for saving passwords, particular charts according to every parameter, list of logins and passwords of registered users for checking by the server and their appearing at the server request. Application Programming Interface (API) performs such functions: execution of the client's request for adding, editing and output of the information; execution of the client's request for receiving information for making a list; execution of the client's request for an authentication, authorization and saving user's password and login; saving client's data and their processing on the server side, saving selected lists, filtration data and sorting; execution of the client's request for editing the contents of database API; execution of the client's request for receiving the information

about the system status; execution of the client's request for receiving necessary information for filtration part.

React and React Native technologies [2], due to which the crossplatform application has been developed, run two different JavaScript scenarios for implementing. The first scenario is designed for raw files finding, choice of them, compilation of the code and making a special collection which is called bundler. The scenario starts with Native-command. While starting the application performs necessary manipulations with the data and the bundler. JavaScript bundler is a classic minifier, i.e. the application to a compiler which uses the code written in the language which is clear for a developer, minifies it reducing the size of the instructions. Such a code is clear only for a compiler but cannot be read by a developer, but it is much faster, and it can be collected into a special bundler which will be performed by JavaScript engine.

Starting the application for work. The files which are gathered into the bundler are placed in the catalogue `/usr/app/src` or `/usr/app/node` modules. Configuration files are situated in the catalogue `/usr/app/app.json`. It is not recommended to edit them manually, for this `expo.config` programme is used. The files `package.json` which are used to run the application and its details are placed in the catalogue `/usr/app/package.json`. The files can also be replaced into any user-convenient catalogue. For this it is necessary to add the way of the `package` file to the configuration `app.json`.

Basic file of `app.json` file configuration looks as it is shown in picture 1.



```
1  "platforms": [  
2    "ios",  
3    "android"  
4  ],  
5  "version": "1.0.0",  
6  "orientation": "portrait",  
7  "icon": "./assets/icon.png",  
8  "splash": {  
9    "image": "./assets/splash.png",  
10   "resizeMode": "contain",  
11   "backgroundColor": "#ffffff"  
12  },
```

**Picture 1.** *App.json file configuration*

Every application user has their own *package.json* file, where it is explained when and what details to include to the application on behalf of this user. For editing the *package.json* file, the special code editor is used, which supports JSON files that allows not interrupting the process of switching on the details while editing. *Package.json* file consists of the objects which are separated with gaps or tabulators. The first object gives the main settings of the application, such as the way to the file where the process of gathering the bundler starts and scripts for starting iOS or Android. The object with application details contains the information about all the details, such as the names of the libraries and their versions. At the particular moment, the libraries connect to the application and perform their functions.

The second scenario is used for interaction with the client on the device side, runs starting of the first scenario, processing user's requests and showing the information chosen by the user.

With the module, crossplatform application has been developed [3]. Owing to the developed module functionality, rapid creating of crossplatform applications has become possible. The module also enables browsing of processed files with API Movie DB and modelling of samples of necessary data according to the particular parameters set by a user [4]. The mechanisms made for the work with a user provide simplicity and flexibility of the work, the simplest and clearest module work interface on a user's level and completely automatic processes of creating of crossplatform functional elements. With the developed software, database work takes place using any devices. The application enables a user to carry out database search and data sampling with the system of filtration.

### **References:**

1. Available at <https://habr.com/ru/company/nix/blog/324562/>
2. Available at <https://reactnative.dev/docs/getting-started>
3. Available at <http://muryj-movie-app.herokuapp.com/>
4. Available at <https://developers.themoviedb.org/>