Serhii Markov

FRONT-END DEVELOPER

SUMMARY

I am a Front-end Developer looking for a full-time position in a company where I'll be able to grow and develop professionally. Motivated to improve my skills and work efficiently on interesting projects. I have knowledge of HTML, CSS, JavaScript, React, React Native. Also, I have hands-on experience in Agile/SCRUM methodology when working in a team. I am a good team player, energetic person, fast learner, responsible and ready for deadlines.

CONTACTS

LinkedIn
email:1nativeflag@gmail.com
tel: (098)2703627

TECH SKILLS

HTML, CSS (LESS, SASS),
Git
React.js
React Native
TypeScript
PWA
Redux
WebSocket
@MUI
Formik
Figma
Next.js
Tailwind css
Classnames
Expo
Mattermost API
Jest
Firebase
Node.js
Express.js
MongoDB

SOFT SKILLS

Industrious
Flexible
Quick learner
Successful work as a team player

EDUCATION

Kharkiv National University of Internal Affairs,
Law-enforcement , majoring in – criminal police,
qualification - lawyer | (2004-2008)

LANGUAGE

Ukrainian — native
Russian — native
English — intermediate

PROJECTS

- **Realtors**
  SPA which implements the ability to search for the purchase, rent, sale of real estate, and it is also a platform for the work of Realtors. This project uses the next technologies:
  React Native, React-hooks, Classnames, Axios,
  react-hook-form, expo, react-native-community/hooks

.

- **OSBB**
  PWA which implements the ability to manage a residential building. Allows:
  users to easily find the information they need;
  reduce energy consumption;
  keep a commercial bookkeeping office;
  share information, discuss and make decisions related;
  control access to the house;
  This project uses the next technologies: React.js, PWA, TypeScript, Redux, Classnames, Axios, react-hook-form, Jest.

- **Chat**
  Allows: the creation of group chats, exchange of messages, ability to search and send contacts, integration with other corporate programs.
  This project uses the next technologies: React.js, PWA, TypeScript, Classnames, Axios, react-hook-form,
  Jest, WebSocket, Mattermost API.

ADDITIONAL EDUCATION

Full stack developer
IT School GoIT
July 2020 - July 2021 |Ukraine

OSBB

This app is designed to make managing your home as simple and efficient as possible. The application offers a wide range of functions that will help control various aspects of ensuring functioning. Key features include:

- Monitoring of energy consumption: the application allows you to view the level of energy consumption of the house and analyze consumption trends over a certain period. This will help you reduce energy consumption and save money.
- Heating management: the application allows you to conveniently manage your home's heating system, ensuring a comfortable temperature and energy efficiency.
- Scheduled Repairs and Maintenance: The app reminds you of scheduled repairs and maintenance so you can complete them on time and avoid expensive repairs in the future, and notifies the community about scheduled and unscheduled events.
- Access control: the app allows you to control access to your home, such as opening a gate or a door lock.
- Interactive notifications: The app sends interactive notifications about new messages and events.
- Information about services and ordering repair or maintenance services directly from the application.
- Advertising goods or services directly to platform users.
- Commercial accounting: The application provides an opportunity to keep commercial accounting of the house, where you can control all financial transactions related to the house.
- Common platform for residents: The application provides an opportunity to create a common platform for residents where they can share information, discuss and make decisions related to life in the house.

This project uses the next technologies: React.js, PWA, TypeScript, Redux, Classnames, Axios, react-hook-form, Jest.

React.js enabled efficient organization and reuse of application components.
Thanks to strict typing, TypeScript improved the reliability of the code.
By enabling offline access, PWA allowed users to use the application without an internet connection.
Classnames provided more flexibility to add classes to elements in code, improving readability and comprehension.
Axios facilitated requests to the server for data retrieval and enabled me to monitor the consumed internet traffic.
React-hook-form efficiently managed forms in the application and optimized form data processing speed.
Jest, used for testing written code, ensured code quality and identified errors before the application was released to users.
In this application, Redux is utilized for efficient and convenient management of the application state (ensuring a single source of truth), simple processing of actions, and a convenient method for restoring the previous state in case of errors.

Realtors

An application that allows you to search, buy, rent and sell real estate. Also, the application is a platform for the work of realtors, which allows them to effectively use the application to find potential clients and conclude deals.

This app has several advantages over other apps for finding, buying, renting and selling real estate:

A wide selection of real estate. The application offers a wide selection of properties for sale, rent and purchase to suit all tastes and budgets, offering not only residential properties but also commercial properties, land plots and other types of properties.

Efficiency. The application allows you to quickly and efficiently find the right real estate and conclude deals, which saves users time and effort.

The platform for realtors. The application provides an opportunity for realtors to offer their services and effectively interact with potential customers.

Intuitive interface. The intuitive interface makes it easy to find the property you need and make deals.

Security. The application provides a high level of security, which is especially important when concluding real estate transactions.

During development, **React Native** was used, which made it possible to create a cross-platform dock for iOS and Android. **React-hooks** are used to work with components, which allow us to effectively work with the life cycle of components. We use the **Classnames** library to manage classes, which made it easy to add and remove classes from components.

The **Axios** library is used to communicate with the server, which allows you to make requests to the API and get the necessary data. To create forms, we use the **react-hook-form library**, which allows you to easily create and validate forms.

Also, **Expo** was used for development, which made it possible to quickly and efficiently develop the application, without the need to install and configure the development environment.

React-native-community/hooks is a React hooks library that has helped to implement certain features in the application faster, reducing the time needed to write the code, reducing its complexity and improving its readability.

This library contains a set of useful hooks such as useBackHandler, useFocusEffect, useNetInfo, useOrientation, usePermissions, useSafeArea, useScrollToTop, useStatusBar, useToggle and others.

They were used for:

- Preserving component state between renderings.
- Running effects after the component is loaded.
- Access to certain functions, such as changing screen orientation, reading network status, working with permissions, etc.
- Support for collapsing the program when pressing the "Back" button.
- Optimizing app performance with various tools like SafeArea, StatusBar, etc.

Jest - used for testing the written code, helped to ensure the quality of the code and detect errors even before the application got into the hands of users.

Chat

The application was created to create group chats, exchange messages, provide access to documents and other resources, the ability to search and send contacts, integration with other corporate programs.

The advantage of using PWA technology is the speed of loading pages, the possibility of using it without connecting to the Internet, the possibility of adding it to the main screen as an application, less development costs, and the possibility of reuse.

This project uses the next technologies: React.js, PWA, TypeScript, Classnames, Axios, react-hook-form, Jest, WebSocket, Mattermost API.

Using **React.js** made it possible to efficiently organize application components and reuse them.
The use of **TypeScript** increased the reliability of the code, thanks to strict typing.
The use of **PWA** allowed users to access the application even offline.
Classnames allowed more flexibility to add classes to elements in code, making it easier to read and understand.
Axios helped to make requests to the server to obtain data for your application and with its help I developed the monitoring of the traffic of the consumed Internet.
React-hook-form provided efficient management of forms in the application and also optimized the speed of form data processing.
Jest - used for testing the written code, helped to ensure the quality of the code and detect errors even before the application got into the hands of users.

The application also used **WebSocket** and **Mattermost API** technologies to provide efficient and continuous data exchange between the client and the server, which ensures high chat performance and reliability.

WebSocket is a technology that allows you to establish a permanent connection between a client and a server. In a chat application, WebSocket can be used to send messages between chat users in real-time. That is when a user sends a message, it is immediately displayed on the side of other users participating in the conversation.
The Mattermost API is an API that allows the chat application to interact with the Mattermost server. Mattermost is a corporate communication system that provides the ability to create closed chats for a team or project. In order to interact with Mattermost, a chat application may use the Mattermost API to retrieve data about users, channels, messages, and other system elements.

OS.community

OS.comunity is an innovative mobile application aimed at swiftly deploying a platform for communication within a closed community. This application aims to unite residents of a specific locality or area, providing them a convenient means of information exchange, coordination, and interaction.
The functionality of the OS.comunity application includes the following features:

1. Chat: This is the primary tool for communication within the community. Users can exchange messages one-on-one or in group chats, enabling convenient discussions on various topics, event planning, and information sharing.
2. Community Events: Users can add and view information about community events. These could include meetings, seminars, festivals, charitable actions, and other events of interest to the community members.
3. Community News: The application offers quick access to news related to the community. Users can learn about local events, receive updates on community-related information, and stay informed about collective initiatives.
4. Creation and Discussion of Community Posts: This feature allows users to create posts on the community's wall, sharing impressions, ideas, questions, and observations. Other participants can leave comments and engage in discussions around these posts.
5. Information about Municipal Establishments and Useful Contacts: The application provides access to a list of phone numbers and addresses of municipal establishments, taxis, restaurants, and other useful contacts. This simplifies the task of finding necessary services and establishments.
6. Searching for Orders for Various Services: Users can search for available orders for various services, such as pet care, plumbing repairs, appliance repairs, and more. This can be useful for those seeking specific services or offering them.
7. Top News Channels: The application allows users to receive the latest news from leading information channels. This helps users stay informed not only about community events but also about global events.
8. Weather Widget: The main screen of the application features a weather widget that provides up-to-date information about weather conditions in the community's region.

During the development of the OS.comunity application, I utilized a stack of modern technologies and tools to create a comprehensive and efficient platform. The development process involved integrating various libraries and frameworks to achieve the desired functionality. Here's a breakdown of the technologies and tools I employed:

1. **React Native**: I chose React Native as the core framework for building the application. This allowed me to develop a single codebase that works on both iOS and Android platforms, saving time and resources.
2. **React Hook Form**: To handle form management efficiently, I utilized the React Hook Form library. This enabled smooth form validation and data handling within the application.
3. **TypeScript**: TypeScript was used to enhance code quality and maintainability by introducing static typing to the project. This helps catch errors early in the development process and improves the overall development experience.

5. **ReactNativePaper:**
   I incorporated ReactNativePaper to achieve a visually appealing and consistent UI design throughout the application. This UI library streamlined the process of creating components that adhere to Material Design guidelines.
6. **react-native-gifted-chat:**
   To implement the chat functionality, I integrated the react-native-gifted-chat library. This library simplified the process of building interactive and engaging chat interfaces within the application.
7. **react-native-vector-icons:**
   To enhance the visual elements of the app, I integrated react-native-vector-icons. This allowed for easy integration of vector icons into various parts of the user interface.
8. **expo-image-picker:**
   The expo-image-picker library facilitated the process of selecting and uploading images within the application. This was particularly useful for features that involve user-generated content.
9. **react-native-custom-qr-codes-expo:**
   I integrated this library to enable the generation of custom QR codes within the application. This functionality can be utilized for various purposes, such as event invitations or sharing user profiles.
10. **react-redux:**
    To manage the application's state in a scalable and organized manner, I incorporated react-redux. This state management solution ensures efficient data flow and synchronization between different components.
11. **Firebase:**
    I leveraged various services from Firebase to enhance the application's backend capabilities:
    - **Authentication:** Firebase Authentication was used for secure user authentication and authorization processes.
    - **Storage:** Firebase Storage enabled efficient storage and retrieval of user-generated content, such as images and files.
    - **Realtime Database and Cloud Firestore:** These real-time databases were employed to store dynamic and structured data, such as chat messages, user posts, and event information.

In summary, OS.comunity is an advanced mobile application that brings residents of a specific region or city closer together by offering them access to information, communication, and services that enhance their daily lives.