Final Project «Advanced front-end technologies»

The Final Project Examination of the Academic program for Major: B057 Information technologies of the 3-year students of the course «SWD2 3307 Advanced front-end technologies». You should build and implement the React App Web based on a topic that you will choose (for example: Social Media App, Drag and Drop List, Messenger or Chat App, eCommerce App, Blog, Books Application, ToDo App, Fitness Appp/Workout Tracker etc.). The purpose of this project is to demonstrate the knowledge of frontend office systems development based on JavaScript libraries and using ReactJS framework. According to syllabus of this course, as well as skills in the most important tools in the React ecosystem: React (including Hooks), Redux, React Router, Webpack, and Babel. This final React App Project requires the implementation of a full-fledged web application that will have a clear goal and provide value for its users.

Notice: The **COURSE POLICY** is - Cheating will not be tolerated. Students caught cheating will receive a "0" for the assignment!

Technical requirements:

1. Install and create an empty project **React App** and choose your own topic. HTML, CSS, and Bootstrap use by default.
2. Use **React JSX syntax** and components, elements depend on your topic of about **15-20 JavaScript files**. Separate components into the external js files and organize your Project structure.
3. Create for React App **functional** and **stateful (class-based)** components by props and states (as components). Use State and **setState()** for working with events handling properties (for example: **onClick, onBlur** and etc.).
4. Manipulate and working with forms for **controlled** and **uncontrolled** components.
5. Use React Fragments. For **CRUD** operations use **setState()** as function for **Add, Delete** and **Edit** elements. Create **Search** and **Filters** components.
6. Choose API tools (**Fetch API, axios,** etc.) for work with **REST HTTP** request and response Server connection. Create **React API components** for API service.
7. Use and make **transforming API data**. Using appropriate methods and JavaScript asynchrony libraries (**async/await**) and functions.
8. Add components for handling API errors. Use corresponding **Error handling functions/methods** and display relevant information to the user.
9. Use React **Lifecycle components** and class-based approach implementation.

10. Create and Use React Lifecycle hook **componentDidMount()**, to perform specific actions when a component is mounted (as lifecycle component stages).
11. Create and Use React Lifecycle hook **componentDidUpdate()** to perform actions when a component is updated or when its props or state change Also create and use Lifecycle hook **componentWillUnmount() for unmounting component** (as lifecycle component stages).
12. Create and Use **React Lifecycle hooks** for error boundaries/handling **componentDidCatch(),** to catch and handle exceptions occurring in child components.
13. Use **React Patterns**, **Pass Functions** as data sources. Use **Render-function** and Implement Clone elements.
14. Implement and use React **High-Order-Components** (HOC)
15. Use and add for component HOC Composition of HOC.
16. Use and manage React **Context API**. Make Dynamic Context switch.
17. Create and implement **React Routes**. Use Routing in your page. Use and add **Relative paths**.
18. Create and use React Hooks. Implement **useState(), useEffect(), useContext(), useCallBack() and useMemo().** Optional: you can create own hooks.
19. Install and add JavaScript library **Redux**. Additional elements with working **React Redux libraries**.
20. Create and connect Redux Components and use **Action Dispatch** for Redux Store.
21. Create Reducer function for Redux Store and use **“Pure function”** components.
22. Use Action Creator and implement **bindActionCreators(),** to generate actions in the Redux Store.
23. Manage **React UI** (User Interface) for **Redux BLL** (Business Logic Layer) and actions with parameters and organize batch for Redux store files.
24. Create React-Redux and function **connect().**
25. Implement function **mapDispatchToProps()** and use as an object.