APPLICATION OF MACHINE LEARNING IN THE DEVELOPMENT OF USER INTERFACES ON THE EXAMPLE OF REACT

Chakvetadze Vissarion Ruslanovich
Senior Software Engineer, Givelify
Tskaltubo, Georgia
E-mail: vchakvetadze@givelify.com

Abstract. The article discusses the aspects of using machine learning in the development of user interfaces using React as an example. A specific research problem, is that even though machine learning has significantly changed all areas of human life, from health care to transportation, a large number of machine learning algorithms are considered “black boxes”, since a large amount of data is entered into it, and the result is a model that can show both maximum efficiency and not be effective at all. A review of literary sources was carried out, which considered the process of development of an information exchange platform for technology transfer management, analysis of methods of development of data forecasting algorithms with the help of machine learning for forecasting cargo transportation, the process of development and implementation of natural language processing methods based on machine learning methods, as well as analysis of application development tools. The analysis of the literature showed that although today there is a large amount of research devoted to machine learning, there is not enough literature on the use of machine learning in the development of user interfaces such as React. Examples of the use of machine learning in the development of user interfaces in the example of React are considered. Research results show that machine learning has made a significant breakthrough in the world and is becoming an integral part of most technological projects, and the development of artificial intelligence and machine learning is expected to lead to significant changes in the development of user interfaces, such as React applications. The conclusion drawn as a result of the research can be useful to programmers who develop user interfaces for React applications.

Keywords: machine learning, React, application, development, interface, artificial intelligence.

Introduction

Even though machine learning has significantly changed all sectors of human activity (Fig. 1), from healthcare to transport, many machine learning algorithms are “black boxes”, that is, a large amount of data is entered into it, and the output is a model that can show both maximum efficiency and turn out to be ineffective at all.

Figure 1 — Use of artificial intelligence in domestic companies by region, %

Also, machine learning has its limitations that must be exploited to effectively use machine learning algorithms. The first limitation considered is the lack of transparency and interpretability since this makes it impossible to track the algorithm for determining the solution by the system.

The second limitation considered is the lack of data availability, as machine learning algorithms need a large amount of data to learn and make accurate predictions.

The third limitation considered is computing resources, however, this limitation can be weakened through the use of distributed and cloud computing, but the cost of the project in this case may increase.

The fourth limitation considered is over-equipping or, conversely, under-equipping. Overfitting is a process in which a machine learning model performs poorly on unknown data because it was trained too well on the training data. However, underfitting occurs because the machine learning model has been significantly simplified.

The final limitation considered is non-causation, which is that correlation does not always imply causation. Therefore, this can significantly reduce the accuracy of forecasting.

Literature review

U.A. Shobanov and S.M. Isaeva [1] in her research analyzed methods for developing data forecasting algorithms using machine learning for forecasting freight traffic; the work examined the process of creating a user application, which consists of the following parts: a user module and a server part.

The work of R.V. Kochetov [2] is devoted to studying the process of developing an information exchange platform for managing technology transfer. It was based on machine learning methods that make it possible to filter data. S.S. Gankovich [3], in his work, reviewed the process of development and implementation of natural language processing methods based on machine learning methods.

The analysis of application development tools was studied by D.G. Nuzhdin, he identified key attributes for comparing popular application development tools and analyzed them according to the identified attributes [4].

It should be noted that although today there is a large number of studies that are devoted to machine

learning, there is not enough literature on the use of machine learning in the development of user interfaces using the example of React, and therefore we can conclude that research in this direction is important and relevant.

The purpose of the study is to analyze the use of machine learning in the development of user interfaces using React as an example.

Methodology

React is a popular open-source JavaScript library (Figure 2) designed for creating front-end user interfaces. Its main difference from other JavaScript libraries is that it is aimed at developing applications through encapsulated units that save state and generate elements of the user application.

It should be noted that artificial intelligence and machine learning are becoming increasingly popular in the field of software development. React uses both artificial intelligence technology and machine learning.

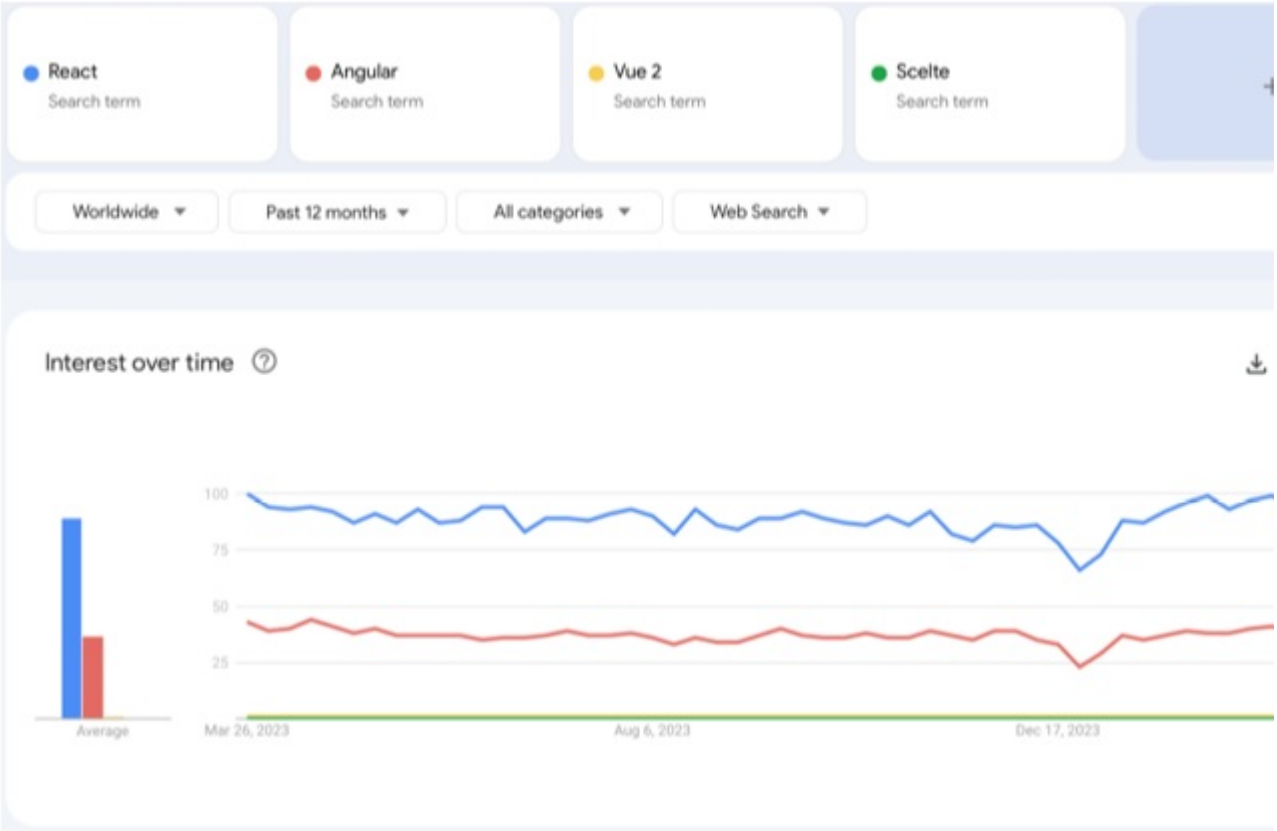


Figure 2 — Popularity of JavaScript libraries for creating custom applications

The analysis will include the following ways that React developers can integrate artificial intelligence and ML into their applications, namely: integrating pre-built AI and machine learning models, creating custom machine learning models, and integrating chatbots. Since ChatGPT is still the leader today (Figure 3), the user interface development process will be discussed using its example.

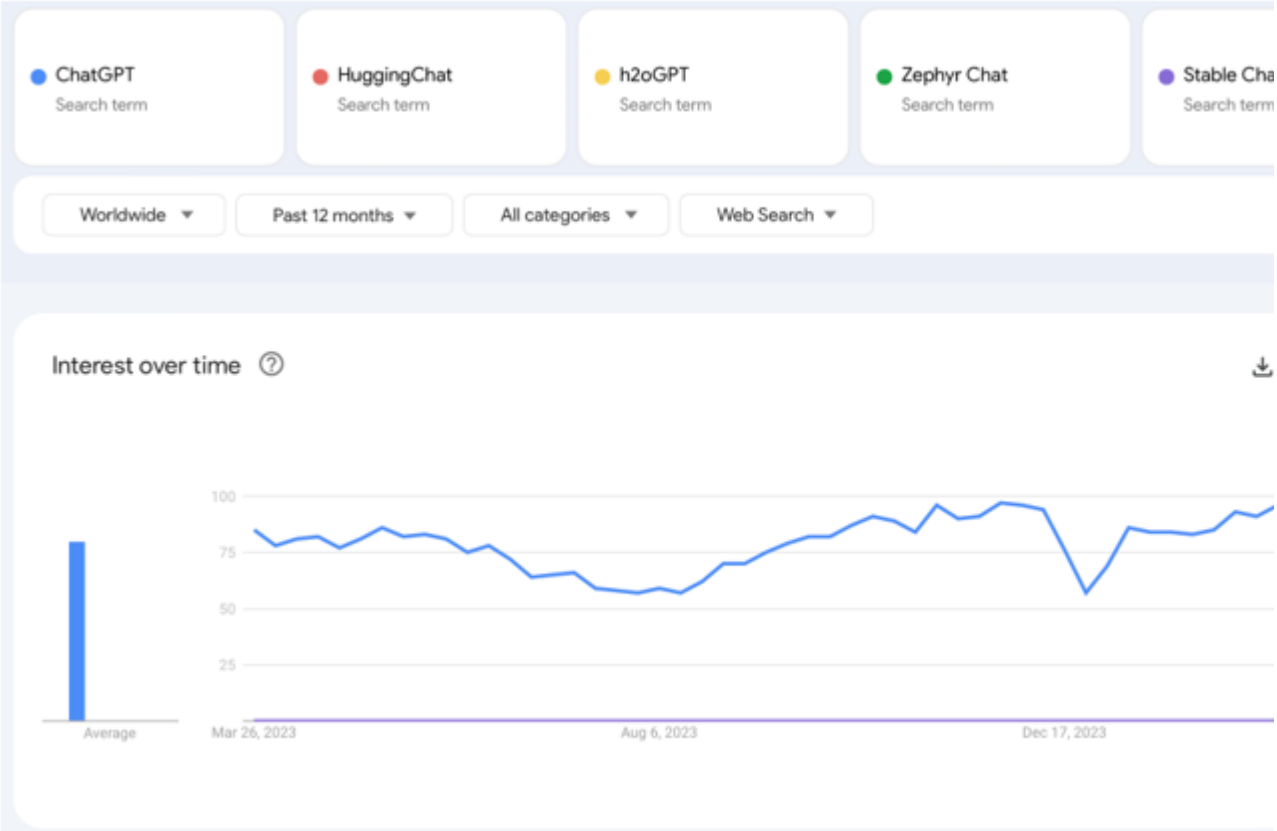


Figure 3 — Popularity of chatbots with artificial intelligence

Research and results

Integration of ready-made artificial intelligence and machine learning models

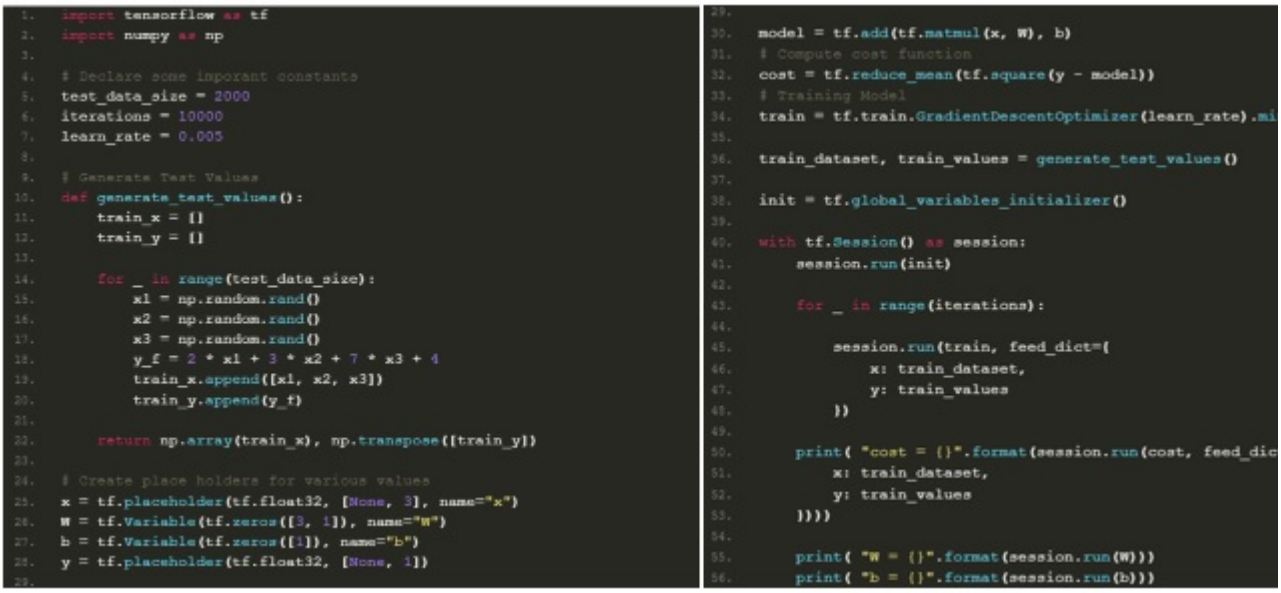
This method is the simplest. Using ready-made models is often more rational than creating your own. There are platforms that offer developers pre-trained models that can be integrated into a Reach application. For example, the Google Cloud Vision API makes it possible to detect objects, faces, and text in images. Rekognition from Amazon has similar properties.

In order to use ready-made models, you need to make an API call on the cloud platform server; for this you can use libraries such as Axios or fetch API.

Create custom machine learning models

Developing your models allows you to train a model for a specific task. An example is the process of training a model to recognize objects or classify images based on their content.

To create a custom model, you will need to use machine learning libraries, such as TensorFlow (Figure 4) or PyTorch (Figure 5). These libraries make it possible to create, train, and evaluate machine learning models. After creating your model, you can integrate it into a React application by creating an API endpoint that returns the model's prediction results.



```python
import tensorflow as tf
import numpy as np

# Declare some imporant constants
test_data_size = 2000
iterations = 10000
learn_rate = 0.005

# Generate Test Values
def generate_test_values():
    train_x = []
    train_y = []

    for _ in range(test_data_size):
        x1 = np.random.rand()
        x2 = np.random.rand()
        x3 = np.random.rand()
        y_f = 2 * x1 + 3 * x2 + 7 * x3 + 4
        train_x.append([x1, x2, x3])
        train_y.append(y_f)

    return np.array(train_x), np.transpose([train_y])

# Create place holders for various values
x = tf.placeholder(tf.float32, [None, 3], name="x")
W = tf.Variable(tf.zeros([3, 1]), name="W")
b = tf.Variable(tf.zeros([1]), name="b")
y = tf.placeholder(tf.float32, [None, 1])

model = tf.add(tf.matmul(x, W), b)
# Compute cost function
cost = tf.reduce_mean(tf.square(y - model))
# Training Model
train = tf.train.GradientDescentOptimizer(learn_rate).mi

train_dataset, train_values = generate_test_values()

init = tf.global_variables_initializer()

with tf.Session() as session:
    session.run(init)

    for _ in range(iterations):

        session.run(train, feed_dict={
            x: train_dataset,
            y: train_values
        })

    print( "cost = {}".format(session.run(cost, feed_dic
        x: train_dataset,
        y: train_values
    ))))

    print( "W = {}".format(session.run(W)))
    print( "b = {}".format(session.run(b)))
```

Figure 4 — Example of using the TensorFlow library

```python
import torch

def MyNetworkForward(weights, bias, x):
    h1 = weights @ x + bias
    a1 = torch.tanh(h1)

    return a1

weights = weights.cuda()
bias = bias.cuda()
x = x.cuda()

y = MyNetworkForward(weights, bias, x)
loss = torch.mean((y - y_hat) ** 2)

loss.backward()
```

Figure 5 — Example of using the PyTorch library

Chatbot integration

Chatbots are another way to use artificial intelligence and machine learning in web development. Chatbots can be used to provide customer support, answer user queries, and perform other tasks. To integrate a chatbot into a React application, you can use ready-made chatbot platforms, such as Dialogflow, or you can create your chatbot using machine learning libraries, such as TensorFlow.

To integrate a chatbot into an application, you need to create an API endpoint that will be able to accept input and return a response to the chatbot. To do this, you can use the Socket.IO (Figure 6) or GraphQL (Figure 7) libraries to manage the interaction between the client and server in real time.

Server

```
const person = {name: 'Rene', age: 26};
socket.emit('person', person);
socket.on('confirmation', () => {
    console.log('The client received the person');
});
```

Client

```
socket.on('person', (person) => {
   console.log(`${person.name} is ${person.age} years old`);
   socket.emit('confirmation');
});
```

Figure 6 — Example of using the Socket.IO library

```
//get all the libraries needed
const express = require('express');
const graphqlHTTP = require('express-graphql');
const {GraphQLSchema} = require('graphql');

const {queryType} = require('./query.js');

//setting up the port number and express app
const port = 5000;
const app = express();

 // Define the Schema
const schema = new GraphQLSchema({ query: queryType });

//Setup the nodejs GraphQL server
app.use('/graphql', graphqlHTTP({
    schema: schema,
    graphiql: true,
}));

app.listen(port);
console.log(`GraphQL Server Running at localhost:${port}`);
```

Figure 7 — Example of using the GraphQL library

We'll also look at how artificial intelligence allows you to create user interfaces for React applications using ChatGPT. ChatGPT is a conversational, natural language, artificial intelligence chatbot developed by OpenAI.

Using ChatGPT, you can create two components, namely: a model window and a button to toggle its visibility. In this case, the request will look like: develop a functional React component that will have a button called ModalOpen. After that, you need to create another component, which will be called Modal with a div element inside.

In this case, ChatGPT will generate the code shown in Figure 8.

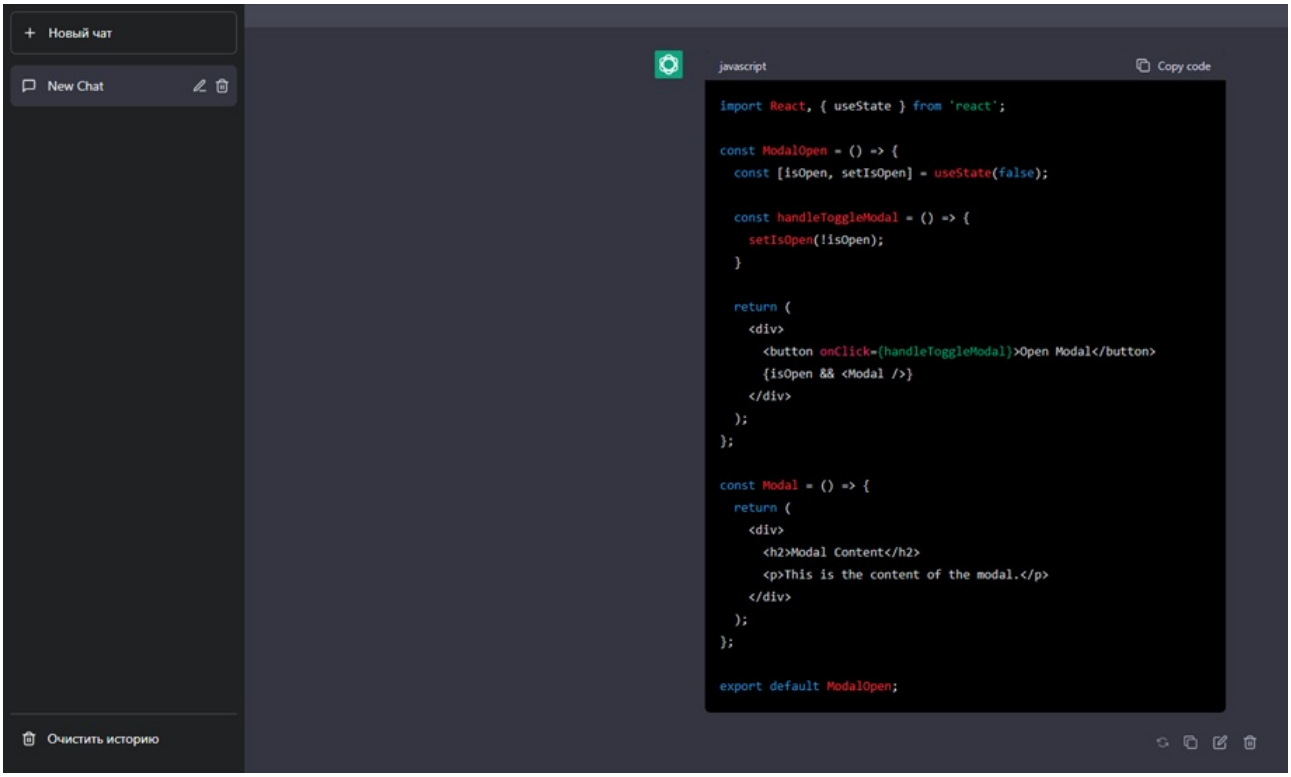


Figure 8 — Code generated by ChatGPT

The next request will be to add functionality, namely: modify these components so that pressing a button opens a modal window (Figure 9).

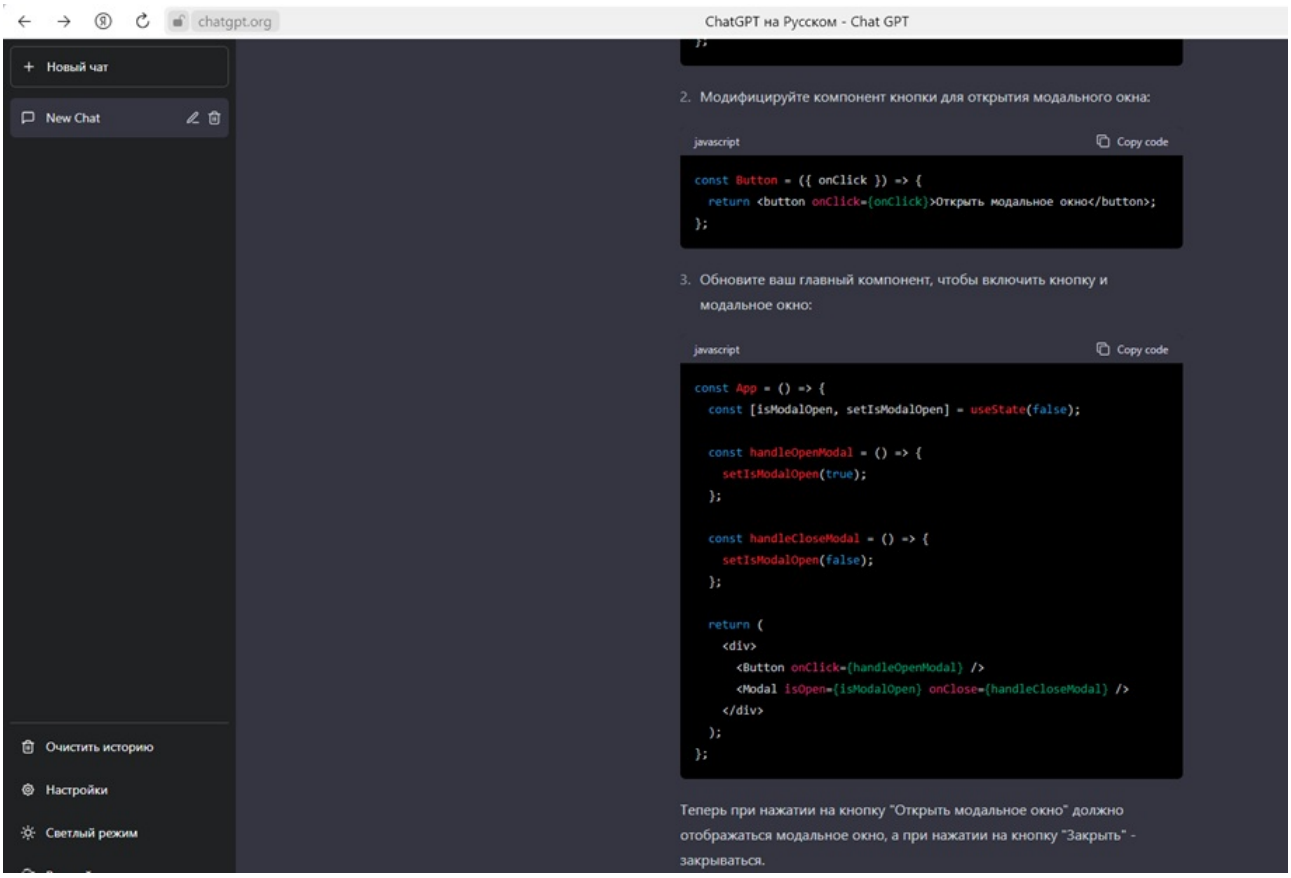


Figure 9 — Code generated by ChatGPT

Let’s add local storage: “Change the ParentComponent so that the isOpen state is stored in localStorage” — Figure 10.

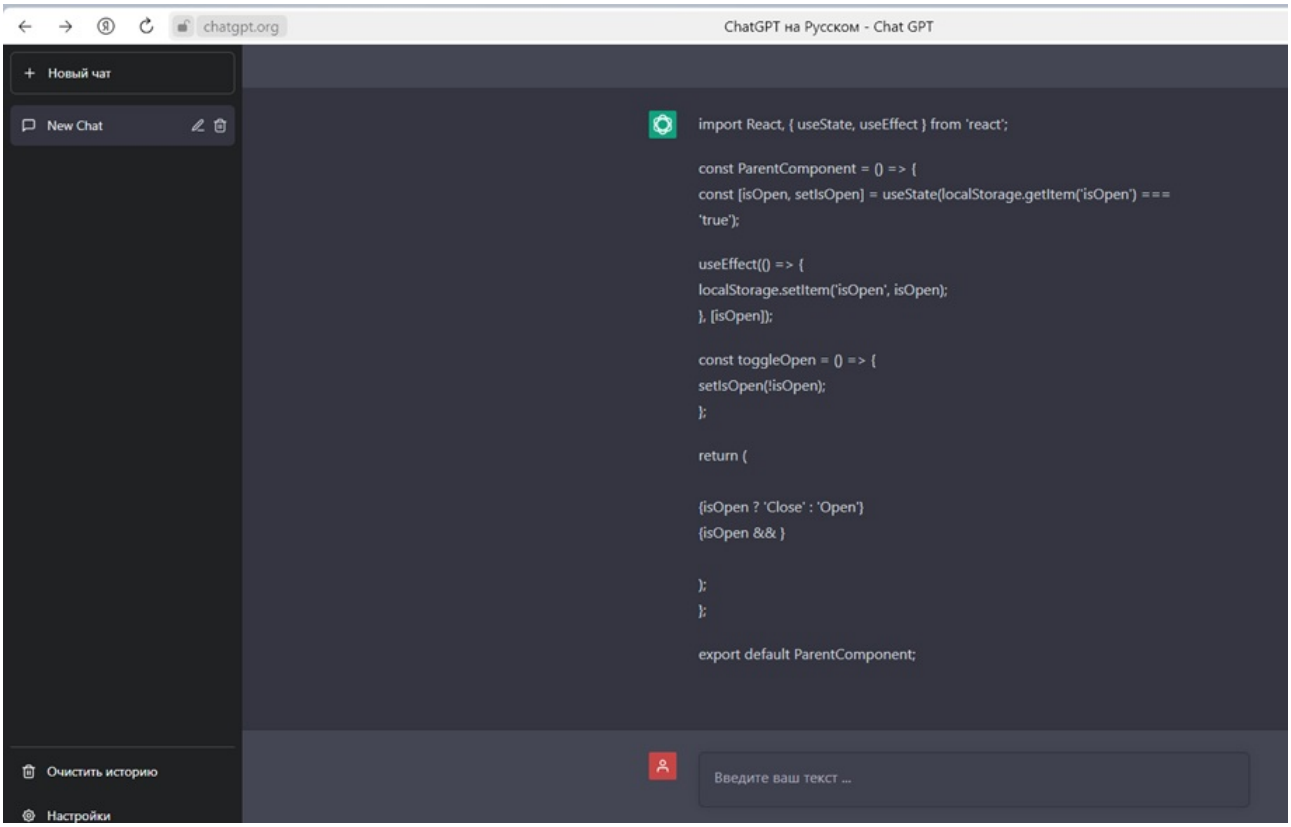


Figure 10 — Code generated by ChatGPT

And let's add ask ChatGPT to style the application — Figure 11.

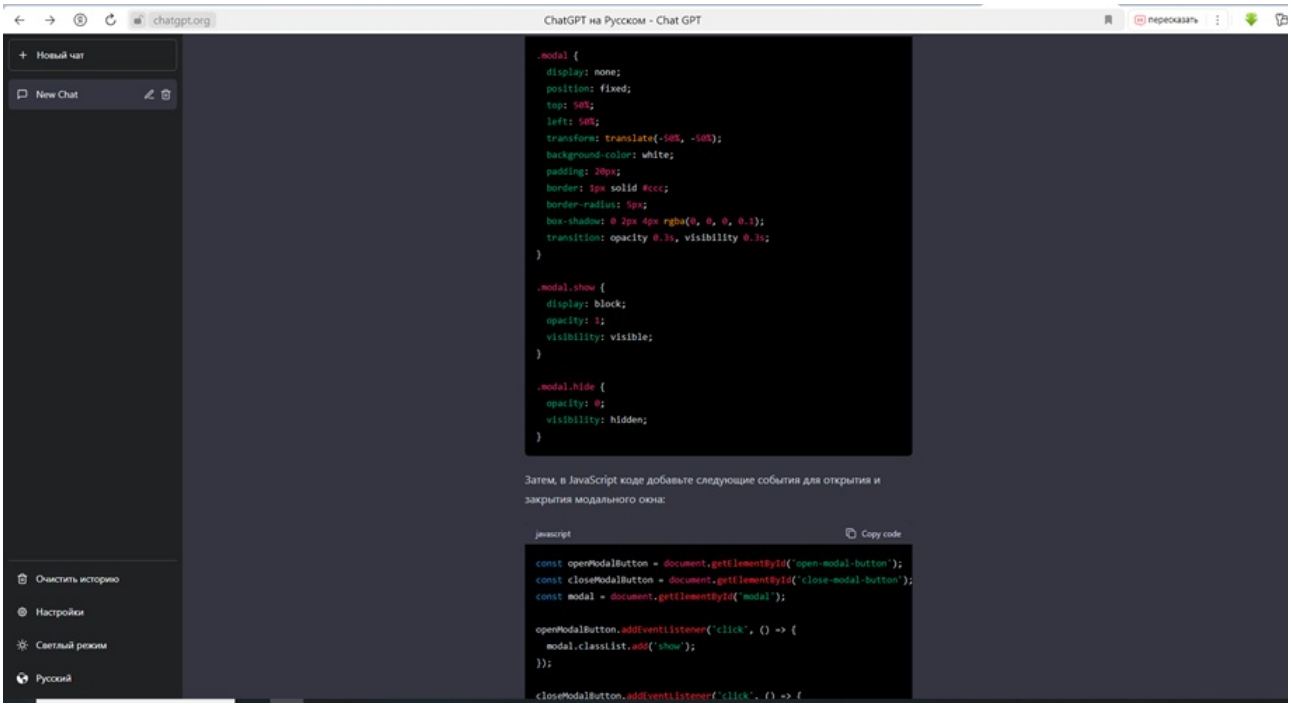


Figure 11 — Code generated by ChatGPT

Thus, ChatGPT fulfilled all the tasks assigned to it, it easily allowed us to implement React components, event handlers, local storage, and styling.

Conclusion

Machine learning has made a significant breakthrough in the world and is becoming an integral part of most technology projects. The development of artificial intelligence and machine learning expects significant changes in the field of user interface development using React applications as an example.

From the above, we can conclude that using machine learning in React applications can allow us to develop more intelligent, interactive and attractive user interfaces. But despite all the benefits, it is necessary to remember the possible security and data privacy problems that the implementation of artificial intelligence can bring.

References

1. Shobanov, E.A. Analysis and methods for developing data forecasting algorithms using machine learning for forecasting trends in cargo transportation / E.A. Shobanov, S.M. Isaeva // Intelligent transport systems. — M.: RUT (MIIT), 2023. — P. 438-444

2. Kochetov, R. V. Development of an information platform for data exchange for managing technology transfer / R. V. Kochetov // Ural Federal University named after the first President of Russia B. N. Yeltsin. — Ekaterinburg: UFU, 2023. — 85 p.

3. Gankovich, S.S. Development and implementation of natural language processing methods based on machine learning methods: master's thesis / S.S. Gankovich // BSU, Faculty of Applied Mathematics and Informatics. — Minsk: BSU, 2020. — 55 p.

4. Nuzhdin D, G. Evaluation and selection of the most suitable tools for developing mobile applications // Universum: technical sciences. — M. 2023. No. 11-1 -S. 37-42.