НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«Київський політехнічний інститут ім. Ігоря Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра програмного забезпечення комп’ютерних систем

##### Практичне завдання №1

**«Запахи коду»**

***з дисципліни "Технології проектування програмних систем"***

Виконала: Родіонова В.О.

#### Група: КП-81мп

Номер залікової книжки: 4212

Оцінка

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1 семестр 2018/2019

**Завдання**

Для виконання даного практичного завдання було взято код з бакалаврського дипломного проекту.

Для аналізу було взято файл DiaryActions.js, що відповідає за маніпулювання записами різних типів у щоденнику.

**DiaryActions.js**

|  |
| --- |
| import moment from 'moment';  import api from '../apiSingleton';  import { formattedDate } from '../utils/dateUtils.js';  import { saveToAsyncStorage } from '../utils/asyncStorageUtils.js';  import { clearSelectedProducts } from './SearchBarActions.js';  export const ADD\_FOOD\_NOTE = 'ADD\_FOOD\_NOTE';  export const REMOVE\_FOOD\_NOTE = 'REMOVE\_FOOD\_NOTE';  export const ADD\_GLUCOSE\_NOTE = 'ADD\_GLUCOSE\_NOTE';  export const REMOVE\_GLUCOSE\_NOTE = 'REMOVE\_GLUCOSE\_NOTE';  export const ADD\_INSULIN\_NOTE = 'ADD\_INSULIN\_NOTE';  export const REMOVE\_INSULIN\_NOTE = 'REMOVE\_INSULIN\_NOTE';  export const GET\_WEEK = 'GET\_WEEK';  export function addNote(date, time, timeType, water) {  return async (dispatch, getState) => {  if (water || getState().searchBar.products && getState().searchBar.products.length) {  const diary = { ...getState().data.food };  const products = (getState().searchBar.products || []).map(pr => ({  id : pr.id,  value : pr.amount  }));  let item = { time, timeType, water: water || null, products };  const id = await api.notes.createNote({ ...item, dateFrom: moment(`${date}T${time}`) }, 'product');  item = { ...item, id };  if (!diary[date]) diary[date] = {};  const notes = [ ...(diary[date].notes || []), item ];  await saveToAsyncStorage('food\_diary', { ...diary, [date]: { notes } });  await dispatch({  type : ADD\_FOOD\_NOTE,  item : { ...item, products },  date,  dataType : 'food'  });  dispatch(clearSelectedProducts());  }  };  }  export function removeFoodNote(date, id) {  return async (dispatch, getState) => {  const diary = { ...getState().data.food };  const res = await api.notes.deleteNote({ id }, 'product');  if (res) {  diary[date].notes = (diary[date].notes || []).filter(item => item.id !== id);  await saveToAsyncStorage('food\_diary', diary);  await dispatch({  type : REMOVE\_FOOD\_NOTE,  date,  id  });  }  };  }  export function addGlucoseNote(date, time, value) {  return async (dispatch, getState) => {  try {  const diary = { ...getState().data.glucose };  const id = await api.notes.createNote({ dateFrom: moment(`${date}T${time}`), value }, 'glucose');  const item = { id, date: moment(`${date}T${time}`).toISOString(), value };  diary[date] = [ ...(diary[date] || []), item ];  await saveToAsyncStorage('glucose\_diary', diary);  dispatch({  type : ADD\_GLUCOSE\_NOTE,  item,  dataType : 'glucose',  date  });  } catch (err) {  console.log('add glucose note error: ', err);  }  };  }  export function addInsulinNote(date, time, value, type) {  return async (dispatch, getState) => {  try {  // const insulinMan = getState().user.info[`${type}InsulinManId`];  const manId = getState().user.info[`${type}Insulin`];  const insulinMan = (getState().user.info.insulinMans || [])[manId];  const diary = { ...getState().data.insulin };  if (!diary[date]) diary[date] = { 'short': [], 'long': [] };  let item = { date: moment(`${date}T${time}`).toISOString(), value, insulinMan, isLong: type === 'long' };  const id = await api.notes.createNote({ ...item, dateFrom: moment(`${date}T${time}`) }, 'insulin');  item = { ...item, id };  diary[date] = [ ...diary[date], item ];  await saveToAsyncStorage('insulin\_diary', diary);  dispatch({  type : ADD\_INSULIN\_NOTE,  item,  insulinType : type,  date  });  } catch (err) {  console.log('add insulin note error: ', err);  console.warn(err);  }  };  }  export function removeInsulinNote(date, id) {  return async (dispatch, getState) => {  const diary = { ...getState().data.insulin };  const res = await api.notes.deleteNote({ id }, 'insulin');  if (res) {  diary[date] = diary[date].filter(item => item.id !== id);  await saveToAsyncStorage('insulin\_diary', diary);  dispatch({  type : REMOVE\_INSULIN\_NOTE,  date,  id  });  }  };  }  export function removeGlucoseNote(date, id) {  return async (dispatch, getState) => {  const diary = { ...getState().data.glucose };  const res = await api.notes.deleteNote({ id }, 'glucose');  if (res) {  diary[date] = (diary[date] || []).filter(item => item.id !== id);  await saveToAsyncStorage('glucose\_diary', diary);  dispatch({  type : REMOVE\_GLUCOSE\_NOTE,  date,  id  });  }  };  }  export function getWeek(date) {  return {  type : GET\_WEEK,  date : formattedDate(moment(date))  };  } |

Даний код має такі проблеми:

1. Дублювання коду – «Замусоварители»:
2. Альтернативні класи з різними інтерфейсами (методи) – «Нарушители ООД»:
   1. addFoodNote, addGlucoseNote, addInsulinNote
   2. removeFoodNote, removeGlucoseNote, removeInsulinNote
   3. методи виконують одні й ті самі функції, але мають по реалізації для кожного типу даних. При додаванні нового типу даних буде необхідно дублювати код.
   4. дані типів: ‘food’, ‘glucose’, ‘insulin’ – можуть бути приведені до однієї структури. Таким чином для додавання чи видалення даних типів можна буде використовувати один і той самий метод
3. Довгий список параметрів – «Раздувальщики»:
4. Довгий метод – «Раздувальщики»:
   1. addFoodNote, addGlucoseNote, addInsulinNote
   2. один метод виконує занадто багато операцій, тому є довгим і складним. На мою думку, метод додавання запису необхідно розбити на такі 3: створення об’єкту запису, додавання об’єкта запису до бази даних (запит на сервер), додавання об’єкта запису до локального щоденника (diary).
   3. А параметри можна передавати у вигляді об’єкту, яким і є набір даних
5. Ланцюжок викликів – «Опутыватели связями»:
   1. const manId = getState().user.info.insulinMans
   2. const insulinMan = (getState().user.info.insulinMans || [])[manId]
   3. Для того, щоб виправити цей недолік, необхідно створити метод getInsulinManByType()

Також, слід зазначити, що деякі методи не обгорнуто у try {} catch () {} блоки, тому можуть спричиняти помилки на більш високому рівні.