// Expenses Reducer

const expensesReducerDefaultState = [];

const expensesReducer = ( state = expensesReducerDefaultState, action) => {

switch (action.type) {

case 'ADD\_EXPENSE':

return [...state,

action.expense

];

case 'REMOVE\_EXPENSE':

return state.filter(({id}) => id !== action.id)

case 'EDIT\_EXPENSE':

return state.map((expense) =>{

if (expense.id === action.id) {

return {

...expense,

...action.updates

}

} else {

return expense;

}

})

case 'SET\_EXPENSES':

return action.expenses;

default:

return state;

}

};

export default expensesReducer;

import uuid from 'uuid';

import database from '../firebase/firebase';

//Add Expense Object

export const addExpense = (expense) => ({

type: 'ADD\_EXPENSE',

expense

});

export const startAddExpense = (expenseData = {}) => {

return (dispatch, getState) => {

const uid = getState().auth.uid;

const {

description = '',

note = '',

amount = 0,

createdAt = 0

} = expenseData;

const expense = { description, note, amount, createdAt };

database.ref(`users/${uid}/expenses`).push(expense).then((ref) => {

dispatch(addExpense({

id: ref.key,

...expense

}));

});

};

};

import {createStore, combineReducers, applyMiddleware, compose} from 'redux';

import thunk from 'redux-thunk';

import expensesReducer from '../reducers/expenses';

import filterReducer from '../reducers/filters';

import authReducer from '../reducers/auth';

const composeEnhancer = window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_COMPOSE\_\_ || compose;

export default () => {

const store = createStore(combineReducers({

expenses: expensesReducer,

filters: filterReducer,

auth: authReducer

}),

composeEnhancer(applyMiddleware(thunk))

//window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_ && window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_()

);

return store;

}