ComputeAnalytics.java

```java
package analytics;

import java.io.File;

public class ComputeAnalytics {

    public static synchronized void computeOrders(HttpServletRequest req) {
        HttpSession s = req.getSession();
        if (s.getAttribute("order") == null) return;
        Order order = (Order) s.getAttribute("order");
        order.computeAllCosts();
    }

    public static synchronized void computeUsersAnalytics(HttpServletRequest req) {
        if (req.getAttribute("users") == null) return;

        HttpSession s = req.getSession();
        Account user = (Account) s.getAttribute("AUTH");
        Map<String, long[]> accounts = (HashMap<String, long[]>) req.getAttribute("users");
        long[] avgTimes = new long[4];

        if (req.getAttribute("allCheckoutAvgTime") == null) {
            req.setAttribute("allCheckoutAvgTime", 0);
        }

        if (req.getAttribute("allAddToCartAvgTime") == null) {
            req.setAttribute("allAddToCartAvgTime", 0);
        }

        if (Utils.isValidUser(user) &&
                !accounts.containsKey(s.getId()) &&
                s.getAttribute("checkoutAvgTime") != null &&
                s.getAttribute("addToCartAvgTime") != null)
        {
            avgTimes[0] = (long) s.getAttribute("checkoutAvgTime");
            avgTimes[1] = (long) s.getAttribute("addToCartAvgTime");
            accounts.put(s.getId(), avgTimes);
        }

        int count = 1;
        for (long[] accountAvgTimes: accounts.values()) {
            long allCheckoutAvgTime = accountAvgTimes[0];
            long allAddToCartAvgTime = accountAvgTimes[1];

            req.setAttribute("allCheckoutAvgTime", allCheckoutAvgTime/count);
            req.setAttribute("allAddToCartAvgTime", allAddToCartAvgTime/count);

            count++;
        }
    }

    public static synchronized void addUserOrders(HttpServletRequest req) {
        HttpSession currentSession = req.getSession();

        Account ac = (Account) currentSession.getAttribute("AUTH");
        List<String> orders = (ArrayList<String>) currentSession.getAttribute("orders");
        String poDir = (String) req.getAttribute("poDir");

        if (orders != null) {
            OrderDAO orderDao = new OrderDAO(new File(poDir));
//          req.setAttribute("orderDao", new OrderDAO(new File(poDir)));
            req.setAttribute("totalAllOrders", orderDao.getPOs().length);

            if (ac != null && Utils.isValidUser(ac)) {
                currentSession.setAttribute("totalOrders", orderDao.getPOs(ac.getUsername()).length);

                for (File f: orderDao.getPOs(ac.getUsername())) {
                    if (!orders.contains(f.getAbsolutePath())) {
                        orders.add(f.getAbsolutePath());
                    }
                }
            }
        }
    }

    public static synchronized void computeAvgStartCheckoutTime(HttpServletRequest req) {
        HttpSession currentSession = req.getSession();
        String url = req.getRequestURI();
        String path = url.replaceAll("/eFoods/", "");

        long checkoutAvgTime = 0;
        int checkoutAvgCount = 0;
        Order order = (Order) currentSession.getAttribute("order");

        Map<String, long[]> accounts = (HashMap<String, long[]>) req.getAttribute("users");

        if (path.toLowerCase().contains("checkout") &&
                req.getParameter("confirm") != null &&
                !order.getItems().isEmpty() &&
                currentSession.getAttribute("checkoutTime") == null)
        {
            currentSession.setAttribute("checkoutTime", new Date());

            if (currentSession.getAttribute("checkoutAvgTime") != null) {
                checkoutAvgTime = (long) currentSession.getAttribute("checkoutAvgTime");
            }

            if (currentSession.getAttribute("checkoutAvgCount") != null) {
                checkoutAvgCount = (int) currentSession.getAttribute("checkoutAvgCount");
            }

            Date startTime = (Date) currentSession.getAttribute("startTime");
            Date checkoutTime = (Date) currentSession.getAttribute("checkoutTime");

            long diff = checkoutTime.getTime() - startTime.getTime();
            long seconds = TimeUnit.MILLISECONDS.toSeconds(diff);

            checkoutAvgCount++;
            checkoutAvgTime = (checkoutAvgTime + seconds) / checkoutAvgCount;

            currentSession.setAttribute("checkoutAvgTime", checkoutAvgTime);
            currentSession.setAttribute("checkoutAvgCount", checkoutAvgCount);

            accounts.put(currentSession.getId(), new long[]{(long) currentSession.getAttribute("checkoutAvgTime")});
        }
    }

    public static synchronized void computeAvgAddToCartTime(HttpServletRequest req) {
        HttpSession currentSession = req.getSession();
        String url = req.getRequestURI();
        String path = url.replaceAll("/eFoods/", "");

        long addToCartAvgTime = 0;
        int addToCartAvgCount = 0;
        Order order = (Order) currentSession.getAttribute("order");
        Map<String, long[]> accounts = (HashMap<String, long[]>) req.getAttribute("users");

        if (path.toLowerCase().contains("api/cart") &&
                req.getParameter("addToCart") != null &&
                order.getItems().isEmpty() &&
                currentSession.getAttribute("addToCartTime") == null)
        {
            currentSession.setAttribute("addToCartTime", new Date());

            if (currentSession.getAttribute("addToCartAvgTime") != null) {
                addToCartAvgTime = (long) currentSession.getAttribute("addToCartAvgTime");
            }

            if (currentSession.getAttribute("addToCartAvgCount") != null) {
                addToCartAvgCount = (int) currentSession.getAttribute("addToCartAvgCount");
            }

            Date startTime = (Date) currentSession.getAttribute("startTime");
            Date addToCartTime = (Date) currentSession.getAttribute("addToCartTime");

            long diff = addToCartTime.getTime() - startTime.getTime();
            long seconds = TimeUnit.MILLISECONDS.toSeconds(diff);

            addToCartAvgCount++;
            addToCartAvgTime = (addToCartAvgTime + seconds) / addToCartAvgCount;

            currentSession.setAttribute("addToCartAvgTime", addToCartAvgTime);
            currentSession.setAttribute("addToCartAvgCount", addToCartAvgCount);

            accounts.put(currentSession.getId(), new long[]{(long) currentSession.getAttribute("addToCartAvgTime")});
        }
    }
}

```