Fundamentals of Software Testing
Assignment Part 2 (of 3)

1 Instructions

- This is the second part of a three-part assignment.
- This assignment will be marked out of 30 and accounts for 30% of the study unit mark. It is estimated that it will take you around 35 hours to complete this assignment.
- The work is to be done on an *INDIVIDUAL BASIS*, not in groups.
- The assignment is to be submitted via VLE by 3rd January 2023.
- A random sample of students will be interviewed about their assignment to reduce the risk of collusion and/or plagiarism.
- Your submission should be a PDF file documenting your efforts and decisions. Code and video recordings should be made available via an appropriate repository (e.g. Github) and file sharing service (e.g. Google Drive) respectively.
- **IMPORTANT:** All questions regarding the assignment should be asked on the assignment forum in the VLE. Questions sent by email will be ignored. Before you ask a question, check if it has already been asked on the forum.

2 Your Tasks

The tasks below build upon the system described in task one of this assignment. Please refer to the assignment specification for more details.

2.1 Task 1: Specify the system

Using your knowledge of the API and the Gherkin (given-when-then) specification (developed in the first part of the assignment) as inspiration, define a specification in terms of a symbolic finite automaton which is suitable for runtime verification purposes.

Note: You are allowed to go beyond the Gherkin specification and you might want to split your automaton into several smaller ones. Please accompany your diagram with appropriate textual explanation of your reasoning, particularly any assumptions and design decisions.

2.2 Task 2: Runtime verification

Using the symbolic automaton specified for Task 1, program and compile a monitor which is able to verify your system at runtime. Document the output of your monitor, particularly by showing how it is able to detect bugs in the system.

Note: Record a video of your monitor execution, upload it to Google drive and share a link to it in your documentation.

2.2.1 Event Log API

In order to help you monitor the the system, we have developed an API endpoint which provides you with a trace of activity on the system. To obtain a the latest traces, you need to make a `GET` request to `https://api.marketalertum.com/EventsLog/{userid}`. This will return a list of events that happened on the system pertaining to that user similar to the following:

```
[
    {
        "id": "3271902f-3184-44cd-b6e6-72ec35abdac0",
        "timestamp": "2022-11-22T06:52:15.2985486Z",
        "eventLogType": 5,
        "userId": "1a90c398-4285-44f1-8ce9-c5efad3d8ca0",
        "systemState": {
            "userId": "1a90c398-4285-44f1-8ce9-c5efad3d8ca0",
            "loggedIn": true,
            "alerts": []
        }
    },
    {
        "id": "c84f894a-e6bd-478d-83f8-4754a066c30e",
        "timestamp": "2022-11-22T06:52:15.4720552Z",
        "eventLogType": 7,
        "userId": "1a90c398-4285-44f1-8ce9-c5efad3d8ca0",
        "systemState": {
            "userId": "1a90c398-4285-44f1-8ce9-c5efad3d8ca0",
            "loggedIn": true,
            "alerts": []
        }
    },
    {
        "id": "9c1e21a2-37cd-4dcc-be03-04a097210b43",
        "timestamp": "2022-11-22T06:52:31.5753157Z",
        "eventLogType": 0,
        "userId": "1a90c398-4285-44f1-8ce9-c5efad3d8ca0",
        "systemState": {
            "userId": "1a90c398-4285-44f1-8ce9-c5efad3d8ca0",
            "loggedIn": true,
            "alerts": [
                {
                    "id": "2ed83c70-9eed-436d-8127-c494926d5602",
                    "alertType": 4,
                    "heading": "Scan Gamer Core i5-12400",
                    "description": "The best gaming PC ever.",
                    "url": " https://www.scanmalta.com/shop/scan-gamer-core.html.html",
                    "imageURL": "https://www.scanmalta.com/shop/scan-gamer-core.jpg",
                    "postedBy": "1a90c398-4285-44f1-8ce9-c5efad3d8ca0",
                    "priceInCents": 31000000,
                    "postDate": "2022-11-21T12:58:42.351779Z"
                }
            ]
        }
    }
]
```

You will note that each event log has the following properties:

id A unique id for the event log.

timestamp A timestamp indicating when the event happened (UTC timezone).

eventLogType The type of event that transpired.

userId The id of the user involved.

systemState A representation of the resulting system state **after the event happened**. The system state indicates the user, whether they are logged in or not, and a list of alerts pertaining to the user.

The possible values of `eventLogType` are as follows:

```
    public enum EventLogTypes
```

```
{
    AlertCreated = 0,
    AlertsDeleted = 1,
    UserValidLogin = 5,
    UserLoggedOut = 6,
    UserViewedAlerts = 7
}
```

In the example above, user `1a90c398-4285-44f1-8ce9-c5efad3d8ca0` has logged into the website (`eventLog == 5`), viewed a list of alerts (`eventLog == 7`) and created a new alert via the API (`eventLog == 0`).

2.2.2 Important considerations about the EventsLog endpoint

It is important to note the following:

- Once you obtain a log of events, it will be deleted from the server. This helps contain memory usage.
- The event log's maximum size is 50 entries. Once the list reaches that size, older entries will be discarded in favour of new ones.
- Similarly, event log entries are only guaranteed to be stored for a maximum of 30 minutes. Any event logs older than 30 minutes will be deleted automatically by the system.

2.3 Task 3: Model-based testing

Using the symbolic automaton specified for Task 1, implement a model-based test suite for your system. Document the output of your test suite, particularly by showing bugs detected and the coverage metrics achieved.

Note: You are allowed to modify the system to intentionally trigger bug detection. Record a video of your model-based test suite execution, upload it to Google drive and share a link to it in your documentation.

3 Dealing with issues

Due to the size of the class and in the interest of fairness, problems and queries related to the assignment should not be handled by email. Instead you are required to:

1. Visit the assignment forum on the VLE.
2. Search to make sure that no one else has asked a similar question.
3. Post your question on the forum.

It is in your best interest to monitor questions posted on the forum by your peers. The system and specification may evolve and **I will consider the forum as a living specification of the system** going forward.

4 Marking Scheme

This part of the assignment accounts for 30% of the marks for this study unit. This will be allocated as follows:

- A correct and comprehensive specification in terms of symbolic finite state automaton - **10 marks**
- A functional runtime verification monitor which generates log output and flags violations - **10 marks**
- A functional model-based testing implementation, including the reporting of suitable coverage metrics - **10 marks**