ECM 1417: Web Development

Assessment Brief (v. 1.6)

Diego Marmsoler

Submission date: Friday 26th March 2021

For your assessment you will develop a *COVID-19 contact tracing* web application. You can get up to 100 marks for the assessment. The assessment is worth 30% of the module mark. The following document describes the requirements of the application in more detail.

The application should be developed in *plain* HTML, CSS, PHP, and JavaScript. Use *only* the frameworks discussed in the lecture. Other frameworks are *not* allowed.

The final submission consists of three parts:

- First, you need to submit all the source files through E-Bart. Do not include the database.

- In addition, you need to submit a one page PDF which
    - Contains the URL of your deployed application which will be of the form:

        `http://<yourmachine>:<yourport>/index.php`

    - Briefly explains if and how you addressed each of the points from the marking scheme described in App. C.

- Finally, you need to deploy your application to your Azure web server. The application must be reachable via the above URL.
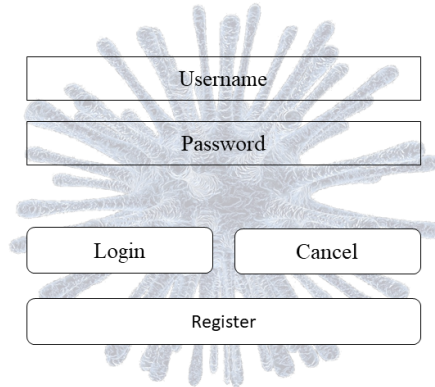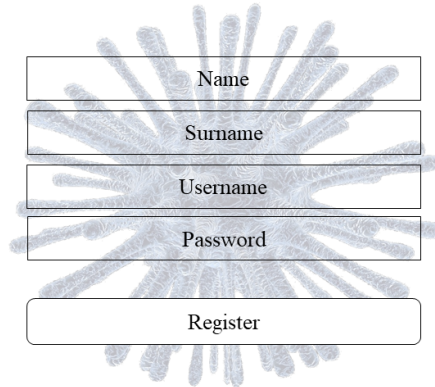

COVID - 19 Contact Tracing

Figure 1: COVID-CT: Login.

1. Login Page

The login screen is depicted Fig. 1. Hitting the register button should open up the registration page (Sect. 2). After successful login, the application's home page should open (Sect. 3). Moreover, a session should be maintained to avoid repeated login requests. Clicking on logout should close the session.

Figure 2: COVID-CT: Registration.

2. Registration Page

The registration page is depicted in Fig. 2. All elements except the surname are mandatory inputs. A password needs to be at least 8 characters long and it may contain uppercase, lowercase, and numbers. The availability of mandatory fields should be checked client side using JavaScript. The password criteria should be checked server side using PHP. To protect the passwords, they should not be stored in plaintext in the database but encrypted using salt (see also Sect. 8). If the registration is successful, the user is automatically logged in and the home page appears (Sect. 3).
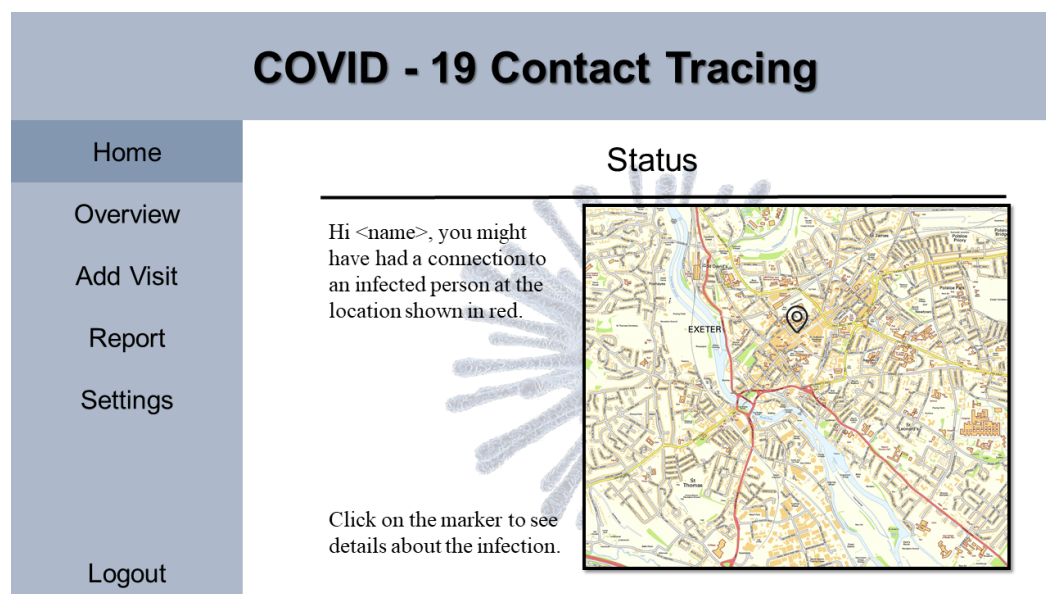
Figure 3: COVID-CT: Home Page.

3. Home Page

The home page is shown in Fig. 3. It should display a personalized contact status of the current user as well as an overview of all places visited by infected persons.

The personalized contact status reports a contact if the current user visited a place within the alert distance (see Sect. 7) which was visited at the same time within the users alert window (see Sect. 7) by a person which reported an infection. For example, lets assume the current user has an alert distance of $x = 5$ and $y = 5$ and an alert window of two weeks. Moreover, lets assume the user visited a location $x = 50$ and $y = 20$ at Thursday 2nd January 2020 from 17:00 till 18:00 and another user X reported an infection on Thursday 9th January 2020 and X visited a location $x = 52$ and $y = 18$ at Thursday 2nd January 2020 from 16:30 till 17:30. Then, the status should report a contact.

The overview is represented as a map of Exeter in which all the places visited by persons which reported an infection within the alert window set by the current user are marked with a black marker. For instance, assume the current user set an alert window of two weeks and another user X just reported an infection. Then, the map should show all the location which user X visited within the last two weeks. The locations which indeed caused a contact with the current user should be depicted using a red marker.

The location of infected persons is obtained from the database as well as the *infections* method of the external web service (see Sect. A).
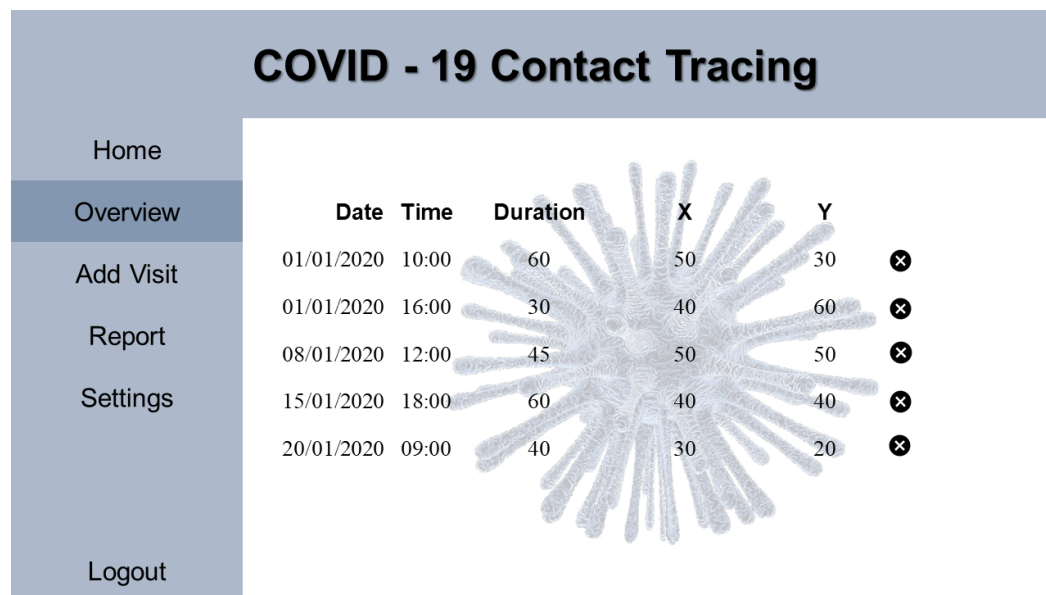
4

Figure 4: COVID-CT: Visits Overview.

4. Overview Page

The overview page is shown in Fig. 4. It should display a table which shows the date, time, duration, and x and y coordinates of each visited location of the current user. Each row should have a remove button at the end which uses JavaScript and AJAX to remove the place from the user's list of visited places.

Figure 5: COVID-CT: Visits Overview.

5. Add Visit Page

The page to add visits is shown in Fig. 5. It allows to input the date, time, and duration of a visit. The coordinates are set by clicking on the map. This should trigger a JavaScript function which places a marker on the corresponding location and writes the coordinates into two hidden form fields which are then send when the user hits add. It should be possible to change the location by clicking again on the map.

Figure 6: COVID-CT: Visits Overview.

6. Report Page

The page to report an infection is shown in Fig. 6. Reporting an infection should store this into the database and also report all the locations visited by the user to the external web service using its *report* method (see App. A).

Figure 7: COVID-CT: Settings.

7. Settings Page

The page to change the user settings is shown in Fig. 7. Settings are stored in corresponding cookies. The user should be able to change two values:

window changes the time window which a user wants to be considered for a possible connection (see Sect. 3). Possible values are one week, two weeks, three weeks or four weeks.

distance changes the distance which a user wants to be considered for a possible connection (see Sect. 3). The distance is interpreted as the maximum Euclidean distance[1] to consider two visits a connection. Possible values are 0..500.

---

[1] The euclidean distance between two points x and y is given by $\sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$

8. Security Considerations

To protect the passwords of your users they should not be stored in plaintext but rather only their hash value should be stored using a random salt for each user. In addition, the web application should be secure against cross site scripting attacks and SQL injections.

Name	Method	Endpoint	Input	Output	Exceptions
infections	GET	/infections?ts={timespan}	n/a	locations	400, 405, 500
report	POST	/report	location	n/a	400, 405, 500

Table 1: REST API of external web service.

A. Web Service API

The Web Service provides two methods to get a list of places visited by an infected person as well as reporting a new location. The Web Service will be available from the beginning of week 09 at the following location:

```
http://ml-lab-7b3a1aae-e63e-46ec-90c4-4e430b434198.ukwest.cloudapp.
                          azure.com:60999/
```

The service's REST API is described in Tab. 1. It lists the name of an operation, the method which can be used to access it, the endpoint to address it, as well as input and output parameters.

A.1. Data

The timespan parameter is an integer which specifies the timespan which should be considered in days from today. For example, a timespan of 5 would return all infections reported within the last 5 days. The payload data is encoded using JSON as described in the following subsections.

A.1.1. Location

Here is an example of a location encoding:

```
{
  "x":50,
  "y":30,
  "date":"2020-01-20",
  "time":"16:00:00"
  "duration":30
}
```

A.1.2. List of Infections

Here is an example of a locations encoding:

```
[
  {
    "x":50,
    "y":30,
    "date":"2020-01-20",
    "time":"16:00:00",
    "duration":30
  },
  {
    "x":80,
    "y":45,
    "date":"2020-12-15",
    "time":"08:00:00",
    "duration":15
  }
]
```

A.2. Exceptions

In the case of success, the request returns a status code of 200. In addition, the following codes are returned to signal an exception:

400 Signals a wrong format for the data.

405 Signals the use of a wrong method to access an operation.

500 Signals an internal server error in which case you should contact me.

B. Style Guide

In the following some additional information about the styling of the page.

- The watermark image is `watermark.png`

- The map is given by `exeter.jpg`

- The markers are given by `marker_black.png` and `marker_red.png`

- The crossmark to is given by `cross.png`

- The main color is given by (red=173, green=185, blue=202)

- The select color is given by (red=132, green=151, blue=176)

- The title of the HTML page is given by the caption of the figure of the page in this document.

- Heading should be rendered in `Arial` size 40, bold

- Menu items should be rendered in plain `Arial` size 24

- Text should be rendered in `Times New Roman` size 20

- Table headings are set in `Arial` size 20, bold

- Table text in `Times New Roman` size 20

C. Marking Scheme

- HTML pages for static content is there and validated (15 marks)
  - Login page
  - Registration page
  - Home page
  - Overview page
  - Add visit page
  - Report page
  - Settings page

- CSS is used to style the pages according to the screenshots and style guide (15 marks)
  - Fonts are correct
  - Page layouts are correct
  - Colours are correct
  - Watermark is correct

- PHP is correct (15)
  - To register
  - To login
  - To check infection
  - To create the overview
  - To add a new visit
  - To report an infection
  - To change the settings

- Secure sessions and cookies (10 marks)
  - Session is created after login and closed after logout
  - Session is destroyed after closing
  - Cookies are used to store the settings

- JavaScipt is correct (10 marks)
  - To remove a location from the overview
  - To select a location from the map

- Ajax is used (10 marks)
  - The corresponding PHP code is valid

- – The Ajax call is correct

- Web Service is used (10 marks)
  - – To report infection
  - – To check for infections

- Security is implemented (10) marks
  - – Salt and pepper is used
  - – Page is secure against cross site scripting
  - – Page is secure against SQL injections
  - – HTTPS is enabled