**Sprint 2**

Eden Dadon - 207279183

Avihay Maman - 205506413

Nofar Elbaz - 316580562

Dudi Biton - 314963141

**Links:**

* Clubhouse - <https://app.clubhouse.io/invite-link/604f752a-93cf-4c19-b570-944bd684cd59> (Invitation links were sent to the lecturer and facilitator).
* Github - <https://github.com/edenvvv/Projects-Management>
* Our website - <https://projects-management-ade.herokuapp.com/>

**Part 1:** (All requirements were entered in the form of a user story in the clubhouse)

**Functional requirements:**

1. The doctor user will have a page with all his details.
2. The doctor user would be able to see his appointments records.
3. The simple user would be able to search for doctors using free text search box.
4. The simple user would be able to filter the results (you should give at least 3 filter options).
5. The simple user would be able to sort the results (you should give at least 3 sorting options).
6. The simple user would be able to rank the satisfaction level of the doctor after this appointment is over (from 1 to 5).
7. The doctor user would be able to send appointment summery to his clients at the end of the appointment.
8. The simple user would be able to see his appointments records.
9. The simple user would be able to book appointments only to doctors who offer services to his clinic.
10. The simple user would be able to see the appointments summery of all his appointments.
11. The simple user would be able to post reviews on the doctor’s private page.
12. The doctor’s reviews would be public to all.
13. The simple user would be able to send his contact information and a note to doctors privately.
14. The doctor user would be able to open public forums where he could answer questions.
15. The admin user would be able to authorize doctor users via email.
16. . The doctor user would be able to delete questions from the forum and manage it at his convenience.
17. The admin user would be able to authorize the new forums after he revied them.
18. The simple user would be able to make appointments only in available time.

**made by us**:

1. The company logo will navigate to home page.
2. The website contains same footer for each page.
3. There will be a back button to the beginning of the home page.
4. The navigation bar will remain on the screen for all over the website.

**Non-functional requirements:**

**made by us**:

1. The system will be secured using the SSL protocol (Quality: Integrity).
2. Adjusting the system for both the computer and the mobile device (Quality: Flexibility).
3. The system will have incredible user experience (Quality: Flexibility)
4. The system will run on any of the popular browsers (Chrome, Safari, Firefox etc.) (Quality: Flexibility).

**Part 1**

**Division of requirements into functional / non-functional requirements & Epics & User stories & Test cases**

(We used the clubhouse tool to perform Part 1 on all its requirements.)

**Functional requirements:**

**# Req 1. The doctor user will have a page with all his details.**

Epic: Details pages.

User Stories:

1. As a doctor i would like to be able to open details page, in order to view my details.
2. As a doctor i would like to be able to edit my details page, in order to add / delete / update my details.

Test cases:

|  |  |  |
| --- | --- | --- |
| **Negative/Positive** | **Test case** | **No.** |
| Positive | Verify that when open the details page, the details will present to the doctor. | 1 |
| Positive | Verify that after clicking on "edit" button, the doctor can edit his details | 2 |
| Negative | Verify that if doctor is not logged in he can access to his details page. | 3 |

**# Req 2. The doctor user would be able to see his appointments records.**

Epic: Appointments.

User Stories:

1. As a doctor i would like to be able to see my appointments records, in order to organize my schedule.
2. As a doctor i would like to be able to edit my appointments records, in order to add / delete / update / delay my appointments.

Test cases:

|  |  |  |
| --- | --- | --- |
| **Negative/Positive** | **Test case** | **No.** |
| Positive | Verify that when open the appointments page, the doctor's appointments will be presented to him. | 1 |
| Positive | Verify that after clicking on "edit" button, the doctor can edit his appointments. | 2 |
| Negative | Verify that if doctor is not logged in he can access to his appointments. | 3 |

**# Req 3. The simple user would be able to search for doctors using free text search box.**

Epic: Perform manipulations on the Doctor's database.

User Stories:

1. As a simple user i would like to be able to search for doctors via search box, in order to decide the best doctor for me.
2. As a simple user i would like to be able to see what I wrote at the search box, in order to check myself.

Test cases:

|  |  |  |
| --- | --- | --- |
| **Negative/Positive** | **Test case** | **No.** |
| Positive | Verify that after click on "search" button, the system will provide the requested doctors to the user. | 1 |
| Positive | Verify that after write at the search box, the text present to user. | 2 |

**# Req 4. The simple user would be able to filter the results (you should give at least 3 filter options).**

Epic: Perform manipulations on the Doctor's database.

User Stories:

1. As a simple user i would like to be able to filter the results, in order to filter out the irrelevant doctors.

Test cases:

|  |  |  |
| --- | --- | --- |
| **Negative/Positive** | **Test case** | **No.** |
| Positive | Verify After selecting a filter, the results will be displayed in the order selected by the user. (x3 – check all filter options). | 1 |
| Positive | Verify that when a user selects a checkbox it is indeed marked with a V. | 2 |

**# Req 5. The simple user would be able to sort the results (you should give at least 3 sorting options).**

Epic: Perform manipulations on the Doctor's database.

User Stories:

1. As a simple user i would like to be able to sort the results, in order to find the doctors who are relevant to me according to my priorities.

Test cases:

|  |  |  |
| --- | --- | --- |
| **Negative/Positive** | **Test case** | **No.** |
| Positive | Verify that after sorting the results, the results will indeed be displayed in a sorted manner to the user.(x3 – check all sort options). | 1 |
| Positive | Verify that when a user selects a checkbox it is indeed marked with a V. | 2 |

**# Req 6. The simple user would be able to rank the satisfaction level of the doctor after this appointment is over (from 1 to 5).**

Epic: Doctor’s reviews.

User Stories:

1. As a simple user i would like to be able to rank the satisfaction level of the doctor after this appointment is over, in order to Allow additional patients to have an opinion about the doctor.
2. As a simple user I would like to be able to add a recommendation about the doctor, In order to provide additional users with more detailed information about my appointment with the doctor.

Test cases:

|  |  |  |
| --- | --- | --- |
| **Negative/Positive** | **Test case** | **No.** |
| Positive | Verify that after ranking the doctor, the results will be public to all users. | 1 |
| Positive | Verify that when a user selects a rank (1-5), he can see his choice on the screen. | 2 |
| Positive | Verify that after clicking on "submit" button, the recommendation will be displayed to the users. | 3 |
| Negative | Verify that not logged user can give a recommendation to a doctor. | 4 |

**# Req 7. The doctor user would be able to send appointment summery to his clients at the end of the appointment.**

Epic: Appointments.

User Stories:

1. As a doctor i would like to be able to send appointment summery to my clients at the end of the appointment, in order to record the meeting.

Test cases:

|  |  |  |
| --- | --- | --- |
| **Negative/Positive** | **Test case** | **No.** |
| Positive | Verify that after clicking on "send" button, mail with appointment summery will be send to user. | 1 |

**Explanation of Part 2:**

In the project we decided to use the following tools:

* For CI / CD we use the circleci tool
* For the CLOUD service for storing and running the project, we chose Heroku (<https://projects-management-ade.herokuapp.com/>)
* For saving the data we chose mongoDB
* The conventions we chose to use are single quotes, no semicolons.



**The software process in this sprint:**

During the sprint running process, we used CI / CD tools when circleci was used to run the tests and dependencies and in addition to checking the conventions we agreed on.

The part of the CD was managed by Heroku.

All requirements were entered in the form of a user story in the clubhouse, each requirement was divided according to a suitable EPIC and in addition received a label for whom the requirement is intended (user, admin, system).

Later in the sprints and at our discretion we will add additional requirements.

We will distribute the requirements among all team members (everything will be managed through the clubhouse).

**Part 3:**

**Division of requirements for sprints**

|  |  |
| --- | --- |
| **Sprint 3 – 60%** | **Sprint 4 – 40%** |
| Requirement No. 1 | Requirement No. 11 |
| Requirement No. 2 | Requirement No. 12 |
| Requirement No. 3 | Requirement No. 13 |
| Requirement No. 4 | Requirement No. 14 |
| Requirement No. 5 | Requirement No. 15 |
| Requirement No. 6 | Requirement No. 16 |
| Requirement No. 7 | Requirement No. 17 |
| Requirement No. 8 | Requirement No. 18 |
| Requirement No. 9 | Requirement No. 19 |
| Requirement No. 10 | Requirement No. 20 |
| Requirement No. 21 |  |
| Requirement No. 22 |  |
| Requirement No. 23 |  |
| Requirement No. 24 |  |
| Requirement No. 25 |  |
| Requirement No. 26 |  |