Milestone 2(Due Sep. 6th):

Split up of tasks:

UI/UX:

Adam : Figma

Frontend (App):

Hoda

API Gateway and external APIs: Hoda, Khalil, Fawzy

Backend and Communication between services:

Khalil

Fawzy

Database and Database management system:

Database management system and database tasks:

Rana

Yasmina

Adam

DevOps and Orchestration:

Fawzy

Tasks:

UI/UX Tasks:

- Adam will create the UX design using figma
- Hoda is creating the front end pages

Backend Tasks:

- Profile management service
    - Sequence diagram
    - Testing
- Patient medical records service
    - Sequence diagram
    - Testing
- Scheduling system
    - Sequence diagram
    - Testing
- API Gateway

Database management system and database tasks:

- Finalize ERD
- Normalize the database
- Implement this in framework
    - Class diagrams

DevOps tasks:

- Deployment diagram

SRS tasks:

- Sprint backlog requirements
- Other requirements

Meetings:

Meeting 30th October:

- Features we will be doing:
    - Users
        - Doctor, Patient, Admin (sudo access)
    - Services
        - Profile management (we could remove it)
            - Create a profile
            - Delete a profile
            - Edit a profile
            - Retrieve a profile
        - Patient medical records
            - Edit an existing record
            - Retrieve a record
            - Delete a record (admin only)
            - Create a record
            - Access management to records
        - Scheduling system
            - Create a schedule
            - Create an appointment in a schedule
            - Edit an existing schedule
            - Delete a schedule
            - Delete an appointment
            - Retrieve a schedule
            - Access management to scheduling
    - Database management
        - Will handle above services
        - Classes
    - Front end
        - Login page
        - Patient home page
        - Admin home page

- Process for SRS (independent or team dependent)
    - Pick the structured natural language requirement you will be doing
        - Make sure it meets the format
    - Put it on Jira
    - Put it in the SRS main document

—--

  - Create the low level design
    - Class diagram
      - Assigning use cases to classes
    - Sequence diagram
  - **Work on it’s implementation and make changes to it accordingly (inputs changed, API calls changed, etc)**

- Process for implementation:
    - Do diagram
    - Start coding
        - Schedule tribe meetings if need
        - Schedule inter-tribe meetings if needed

Agreed upon format for requirements:

2.3

Scenario: After an appointment or before an appointment with a patient, a doctor would like for them to run some tests so they look up the patient and send a request for the desired tests.
Function: A doctor can request a test for a patient.
Description: Through the **prescription management system** a doctor can request a test for a current patient which is then reflected in the patient's prescriptions.
Inputs: Doctor ID (string); Patient ID (string); the test requested (string).
Source: The client request source is from the **API Gateway**; the doctor requests the test through the Doctor UI.
Outputs: The requested test for the patient with reference to the doctor who issued it.
Destination: The patient's prescriptions are updated through the **Database management system**. The patient is notified of the test request through the **Notifications API**.
Action: If a patient is scheduled to have a visit to a doctor or has had a visit the doctor can request a test be added to the patient prescriptions.

Requirements and Pre-condition: The system must have at least one organization with a doctor profile, at least one patient who has a scheduled appointment or has had one.
Post-condition and Side effects: The test is added to the patient's prescriptions, but the patient can choose to reject the request and remove it directly using the **prescription management system**. A notification is issued to the patient indicating the test request through the **Notifications API**.

Milestone 1(Due Oct. 23rd):

Product Backlog + Requirements gathering (Till Oct 10 [Thurs])

Throughout the week: work on backlog individually
Thursday 9pm: Discuss the backlog + plan for architecture
Weekend: individual work

Architecture (Till Oct 13 [Sun])

Meeting Sunday 1:30 meeting to integrate the design
-> Go to dr's office hours

Use Cases (Till Oct 16 [Wed])

Requirements Document (Till Oct 20 [Sunday])

Presentation (Presenting on Wednesday, Wed 23)

Tasks:

When working on 1 or 2 add what you did under what you worked on for others to know

1. Use case
    a. Add actions from the backlog (ignoring subsystems for now)
        i. Adding the connections

Done: a preliminary diagram of the actions to be discussed tom with the TA

2. Architecture
    a. Layered Architecture
    b. Microservice architecture
3. Requirements document
    a. User requirements
    b. System requirements

c. Dr has an example in the project document
d. Jira
e. Specify our -ilities
f. What features we plan to actually implement
g. Are the non-functional requirements testable?
h. Tech stack

Meeting 17th Oct topic(s):

- Discord (Done)
- Presenting architecture and use case and taking questions (Done)
    - Could stay as services and still be microservices
    - Considering them to be components instead of services
        - Could become fully layered

- Planning and splitting tasks for the requirements document Task
    - Structured natural language
    - Linking between user and system requirement suing numbering
        - Used in github commits
        - Explaining this somewhere in the document
    - Jira

- Start pushing stuff onto GitHub
- Discussing ERD
    - Ask working Drs about more things we need from the actual medical field

Questions for dr

- Are we supposed to show replication in our diagram for availability?

Meeting 22nd Oct:

- Make sure everyone understands architecture
- Discuss and review SRS
    - Fix numbering
    - Decide on unified format (right now the document looks like it was written by 6 different people lol) Because it was :)
- Check that all milestone requirements have been satisfied

    - We still haven't picked out the subset we'll cover, covered the risks or created a design specification document(?)
    - What else have we missed?
- Discuss submission details
- Make sure everything is on github
- Plan and split up presentation
    - Everyone has less than 3 mins to speak

1.Overview of your proposed SW system
(What it is about
The users
The motivation behind it). (Hoda)

2. The product backlog which will include all the requirements you can think of
from start to end (including functional and non-functional).
Patients = 1.X (Rana)
Doctors = 2.X (Adam)
Administraives = 3.X (Khalil)
Pharmacists = 4.X (Khalil)
Nurses = 5.X (Khalil)
Emergency Responders = 6.X (Khalil)
Non-functional = 7.X (Hoda)

2.1 Your requirements must tackle issues related to global, cultural, social, environmental, economic, public health, safety, and public welfare.
2.2 An identification of the features that you will be completely implementing
throughout the course (which could be a subset of the full product backlog).

4. A set of use cases using use case diagrams. (Yasmina)

3. A high-level architecture of the system based on your product backlog, which
must be done using UML component diagrams. (Ahmed)
    Database (Rana)

5. Any identified risks.(Yasmina)

9. Conclusion (Khalil)

Use case diagram:

Services:

1. Scheduling
    a. Schedule appointment [includes by rescheduling]
    b. View appointment
    c. Payment
    d. Set the schedule
2. Update Medical History extends/includes View Medical History
3. Update Profile
4. View Profile
5. View Prescriptions
6.

Architecture:

- Authentication and Authorization
    - Authentication Service: to verify the identity of users when they try to log in.
        - Accepting login requests

        - Verifying credentials
            - Password check, username check, …
        - Generating necessary tokens
        - Handling multi-factor authentication (MFA)
    - Authorization Service: To control what the authenticated users are allowed to do based on their roles/permissions.
        - Managing roles and permissions (doctor, nurse, patient, admin, pharmacist, emergency responder)
        - Checking if the user’s token grants them access to specific resources (medical records, prescriptions, …)
    - User Management Service: To handle user registration and user profiles
        - Managing user signup, updates (profile management), and deletions (account removal)
        - Hashing passwords
        - Handling password recovery

- User Interface Management
    - Session Management Service
        - Handling login sessions by storing or managing tokens
        - Managing session expiration and refreshing tokens when necessary
        - Keeping track of active user sessions and ensuring users are logged out when needed
    - Role-Based UI Service
        - Customizes the UI experience based on the user’s role (patients, doctors, emergency responders, pharmacists, admin)
        - Adjusting the user interface according to the role assigned by the authentication/authorization layer
    - Notifications Service
        - Sending role-based notifications to users .
        - Displaying real-time updates (to schedules, prescriptions, payment)

- Business Logic: I think you can tell from the names

Possible Project Ideas

Online Bakery Shop (Website)

Summary: Designing a website for an existing bakery would allow us to effectively practice requirements analysis, designing, developing, etc., a product for an actual client, with a likely chance that the website could actually be used by real customers.

Possible Features:

- Different tabs for different sections (cakes, cookies, savory items)
- Customization details for each order (size, number) => according to the business requirement
- Carts to save orders
- Ordering page (address/details/etc.)
- (Email?) confirmation for orders
- A method to notify the owner of the new orders
- A method for the owner to update the roster of the items
- Whatever other features align with the business needs

AUC Gap (App)

Summary: Since everyone's schedule here is unique and most likely not aligned with those in their friend groups, it's often a hassle to figure out who's free and where they are during a gap. Thus, this app would be away for people to share their location within the AUC, along with some extra details, to a select group of people

Features:

- The ability to add friends/ remove them/ hide location from certain people
- Having a status bar to indicate if the person is busy or is welcoming for people to join them
- The ability to create a "group meeting" that shows multiple people meeting together in the same location
- The locating feature could be using geo-location, or manually entered by the person (for privacy/ technical reasons)
- The ability to turn off the location when needed

AUC Talabat (need a better name)

AUC Everything

AUC Help (requesting app)

Egypt Transportation

Calculations checker

It is an app that will check for simple calculation errors in a long mathematical solution just like how a spell checker checks for spelling mistakes. For example, when someone is solving a really long mathematical problem sometimes they make simple calculation mistakes that mess up the entire problem ,causing them to resolve the problem multiple times. If they had an app on their ipad to correct the calculation mistakes they make as they solve the problem they wouldn't have to solve it more than once. The app could also have a feature to help people that do not use ipads where they can scan their solution and the app checks if there are any calculation mistakes. The calculation checks will only be for simple calculations such as arithmetics or simple derivatives to prevent cheating. Another feature in the app could be an immediate calculator where if you write any arithmetic operation followed by an equal sign then the app will calculate it and place the answer after the equal sign. You can also select the answer you get and turn it into scientific notation or convert a decimal number to a binary or hexadecimal.

Development Tools

Full Stack:

Flask:

- Uses python
- Easy enough to use
- Full stack development: frontend (js), backend(python), and our choice of Database
- Only web applications

Front-end

Flutter:

- Easier to learn (apparently javascript is very tricky)
- Simple to make cross-platform applications
- Flutter is more fun (subjective but I agree)
- Firebase
- Best for apps => probably not so much for web

React:

- More relevant in the job market
- Better for web development
- Better for search engines
-

Other options: angular, SvelteKit

Useful Links:

- Flutter vs Angular vs React: The Ultimate Showdown [2024].
- Coming from python, flutter over react native? : r/FlutterDev
- flutter vs react native what is better to learn.. : r/FlutterDev
- flutter vs react native what is better to learn.. : r/FlutterDev