CSCI 265 Project Design (Phase 3)

Team Name: WeWantGOODMarks

Project: C-Clo

Contact - Daniela Malagon danielitamalagon@gmail.com

Table of Contents:

Known Issues/Omissions:

- Regarding User Security and Privacy; the User's Information will be encrypted and passwords will be stored using hashing algorithms. This may be difficult to implement and may be an issue in future implementation and design.

Product Overview:

The vision for this project is to create a user-friendly, informative period tracker website. If the user wants to keep the submitted information stored and possibly receive notifications, they can create an account and login whenever they visit the website. The website's main features are: the period tracker, various health/hygiene information and user accounts.

For the project, we decided to go with an object-oriented logical design. We split it into three main subsystems: the User Interaction subsystem, the Trivia subsystem and the Accounts subsystem.

Under the user interaction subsystem, we have the Moods, Symptoms and Calendar pages. All of these pages take in a user input and produce an output. The Moods page produces some advice, the Symptoms page produces possible medication (based on the input symptoms) and the Calendar page produces a prediction.

Under the trivia subsystem, we have the Health and Education pages. These two pages simply have information on various hygiene topics and any issues someone may encounter.

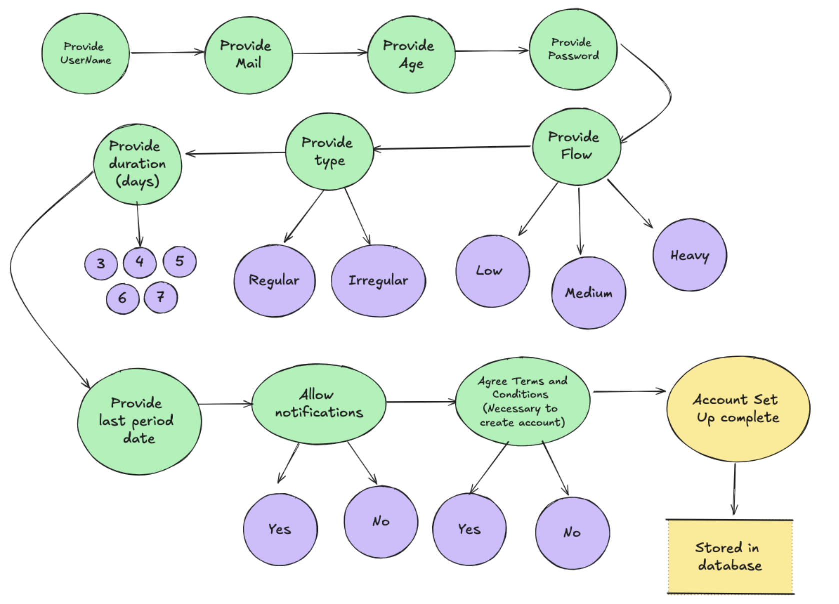
Under the account subsystem, we have the account set-up and user accounts. This subsystem covers all things related to accounts. This will likely be the most difficult aspect of this project.

To implement this project, we plan to use HTML and CSS for the web pages and Javascript for the period tracker. We plan on using Javascript and MySQL databases for the user accounts.

Logical Design and Preliminary Thoughts on Transition:

---

1. **Setting up an account/Registration page:**



This is the account registration page and it is the first page the user encounters when entering this website. This page gives the user the option to sign up to the website so that they may keep their information tied to an account and receive notifications (in the form of an email) to warn them of upcoming period dates.

To create an account, the user will have to provide an email address, their age (for certain content purposes and the possible stretch goal of a chat system), as well as their period information to give them a calendar immediately. The user will have the choice to receive notifications. They must also agree to the Terms and Conditions of creating an account. Once all of this is complete, the information is stored in the database to let the user access it at any time. For more details, refer to DFDSA.

1.1 Process:

- User Accesses registration page: The user lands on the webpage, once opening the site the registration page is clearly shown.
- Initial Data Collection: User fills out required fields:
    - Email address
    - Age
    - Period Information(Flow,type,duration…)
    - Notifications(yes/no)
- Terms and Conditions: The user must agree to the Terms and Conditions or else the account won't be created
- Submit: Once the above has been completed, the user can click the “Sign up/Create Account” button, the information is validated.
- Data Storage: if the user input is valid, the data is securely stored in a database
- Confirmation (stretch goal on how detailed this will be): The user possibly receives an email thanking them for creating an account and in the email a jumpstart guide to the website.
- Access to Account: The user can now log in access their account and view notifications

1.2 Data flow:

- Input Data Flow:
    - User inputs email, age period data, notification preferences
- Validation Flow:
    - The system checks if user input is valid
- Notification Flow:
    - The system sends upon registration, a notification email to the user
- Data Storage Flow:
    - Correct and valid data is stored in the database

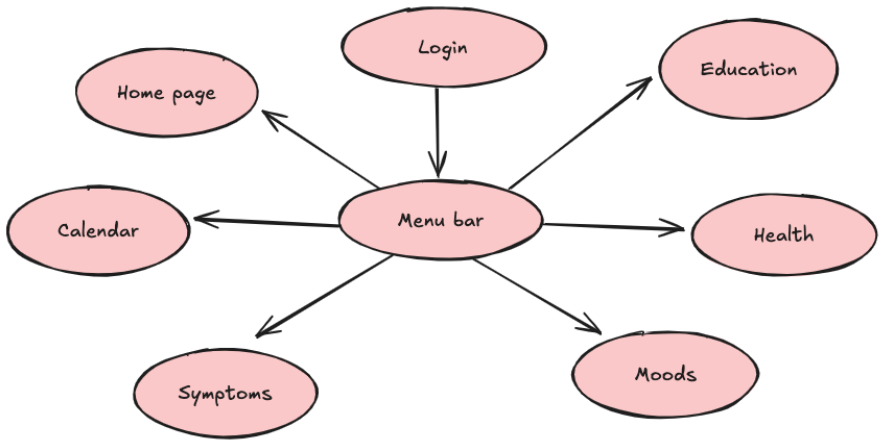
1.3 Data Stores:

- User Database:
    - Stores User login material(email, password, age)
    - Stores Period information (flow, type, duration, last period date)
    - Stores Notification preferences
- Logs Database (stretch goal):
    - Tracks login attempts, like successful and failures, for security purposes

1.4 Preliminary thoughts on the transition from the logical design to an implementation:

For the implementation of the Front-End, we will use HTML and CSS, for Back-End Javascript, for Database MySQL. For security considerations, we plan to use proper hashing for any sensitive information like passwords and on the database side comply with relevant data privacy laws. Our end product to users should be a clean, user-friendly, and easy-to-use website login page.

2. System context:



Once the registration/account sign-up is completed, the user will be directed to the Home page. The home page provides an overview/dashboard of the features. The navigation bar in the home page will connect to most of the features of the period tracker. The navigation bar will include options such as Calendar, Symptoms, Moods, Heath, Education and more . The user will be directed to the corresponding page they'd like to visit by clicking on these options. For a clearer visual representation please refer to DFD0.

Processes:

Input: Users can click on one of the menu bar options, each corresponding to one of the features of the website (example: Education, Health, Moods, Symptoms, Calendar..)

Process: The Home page processes the selection of the user and identifies the suitable section to navigate.

Output: The user is redirected to the selected page.

The processes, such as navigation and menu bar options, do not require traditional data input or output. They mainly function through navigation. Actual data input/outputs occur when the user enters a particular page.

2.1 Data Flow

Since the Home page is just a Navigation hub with no data, most of the actual data flow happens in the section pages.

In addition to providing a menu bar to navigate the features, the Home page also displays the user's current cycle date. This data (the current cycle date) is retrieved from the period calendar of the user.

From the Calendar section to the Home page:

Description: Current cycle date retrieved from period calendar.

Source: Calendar data.

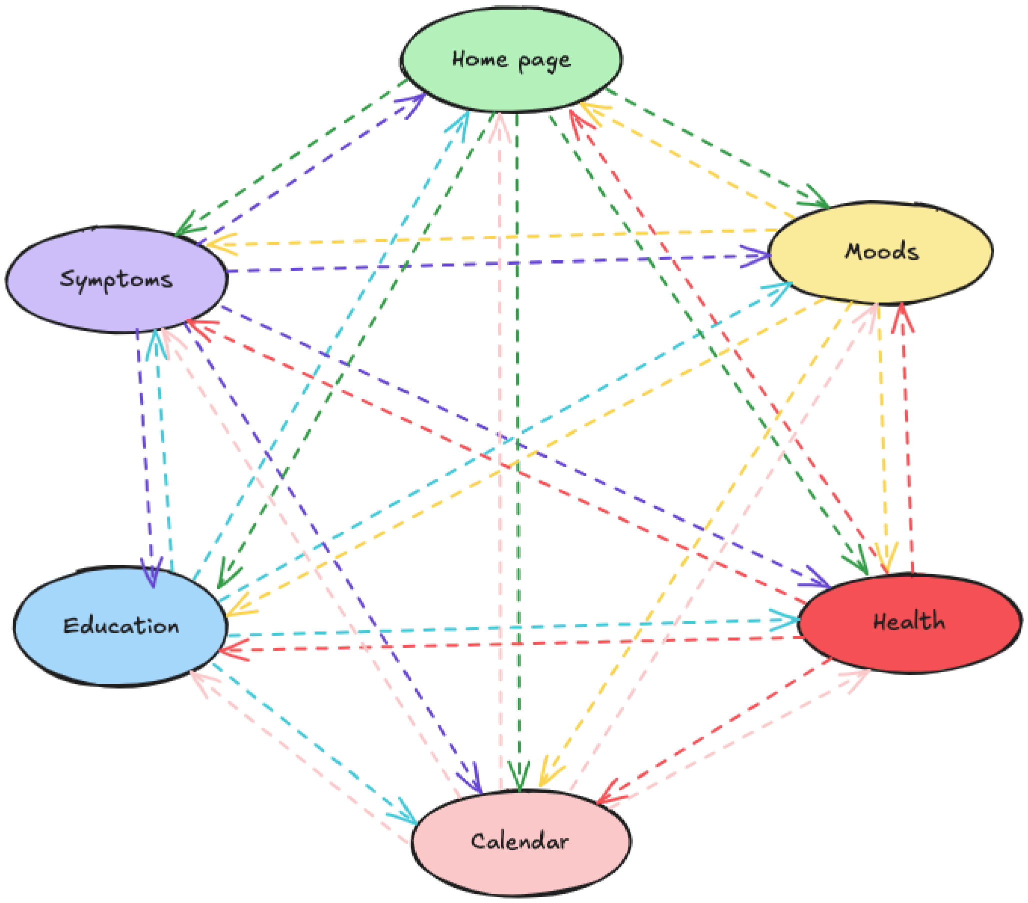
Destination: displays on the Home page.

2.2 Data Stores:

There is no direct data storage for the home page, instead, it displays the data retrieved from the Calendar section when a user accesses it. The primary role of this interconnecting page is to route to different sections/features of the website. The data input and output occur within the selected pages(for example: Calendar, Symptoms, Moods), while the home page simply facilitates access to these pages.

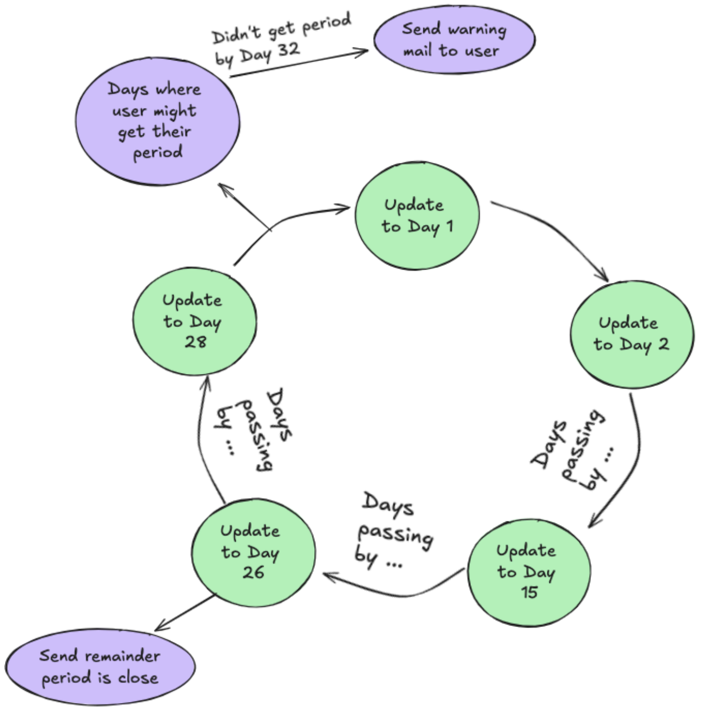
2.3 Preliminary thoughts on the transition from the logical design to an implementation:

Our project will utilize HTML, CSS and JavaScript to implement period tracking. The Homepage will facilitate navigation through different sections/features of our website. HTML can handle the basic navigation requirements without additional technologies. This will help users to move around pages easily.



The chart DFD1 shows our website's flexible and user-friendly navigation. Features of the website's home page are clear and organized in the navigation bar, making it easy for users to access the Calendar, Moods, Symptoms, Resources and Education pages (with a click). Users can smoothly transfer from the Home page to a selected feature page. Once inside any selected feature page, they can go back to the Home page or any other feature page.

3. Home page module:



This is the main page where the user can see what day they are currently in. It provides the date of the user's upcoming period, and some quick links to easily add a date or ask for advice on their symptoms or moods. This is a system that works as a cycle where the date is updated until the user gets their period, then the cycle will be restarted (updated to Day 1). The data flow diagram (DFDMP) is shown above.

3.1Process:

Description:

This process handles updating the day as time passes and as the user updates their period starting date. By Day 26, the user will get a reminder (email) that their period is coming. If a date hasn't been added by Day 32, another notification will be sent to the user.

Detailed descriptions of the process sequencing can be seen in the sequence diagram DFDMP

Inputs:

Data (incoming): Dates provided by the user

Outputs:

Data (outgoing): Current day of the user.

3.2 Data flow:

Description: This represents the information being passed from the calendar regarding their period starting date.

Source: User (external entity)

Destination: Main page process

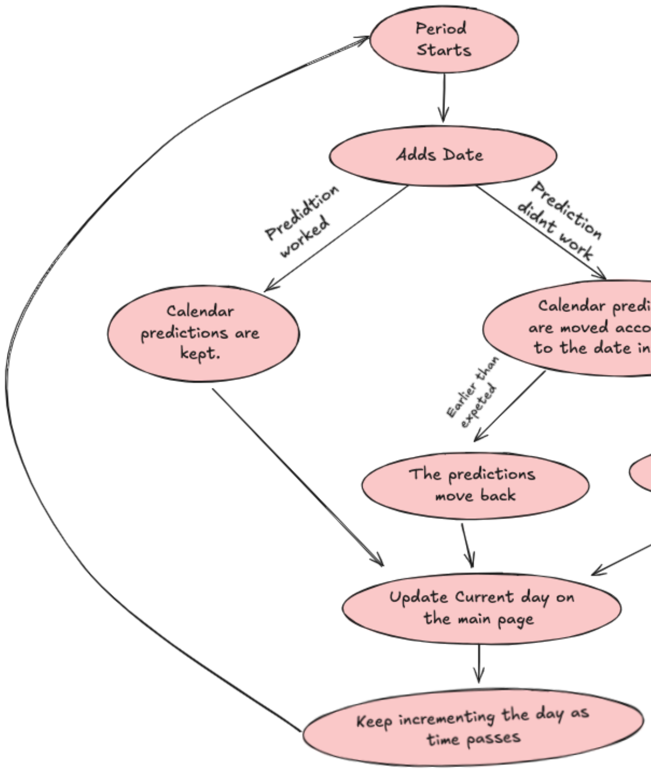
3.3 Data stores:

The data of the period starting date is stored under the user account in the database. Based on this information, the displayed date on the main page will be updated.

3.4 Preliminary thoughts on the transition from the logical design to an implementation:

The programming languages that are going to be used are HTML, CSS, and JavaScript. HTML and CSS take care of the graphic elements, and JavaScript will take care that the dates are updated correctly, meaning there are going to be three files for this menu option. These coding files will be placed in the development branch with the menu option as its name. This is the only menu option that presents a connection with another menu option, therefore, proper connections for networking will be made.

4. **Calendar Module:**


Period Starts
Adds Date
Prediction worked
Prediction didnt work
Calendar predictions are kept.
Calendar predictions are moved accordingly to the date inserted
Earlier than expeted
Later than expeted
The predictions move back
The predictions move forward
Update Current day on the main page
Keep incrementing the day as time passes

This is the calendar page, where the project's main output from the period tracker is found. Once all required inputs are submitted by the user, the calendar page will display a 6-month calendar with possible dates of the user's period highlighted. The closest date will be displayed on the main page as discussed in the above module. This calendar can be changed/updated at any time. The data flow diagram (DFDP) is shown above.

4.1 Process:

Description: The user fills out the period tracker questions. Once submitted, the page will display a calendar.

Inputs:

Data (incoming): Flow type, regular/irregular, duration, last period date.

Outputs:

Data (outgoing): A 6-month calendar with predicted dates based on user input.

4.2 Data flow:

Description: This represents the information being passed from the user regarding their period.

Source: User (external entity)

Destination: Calendar process

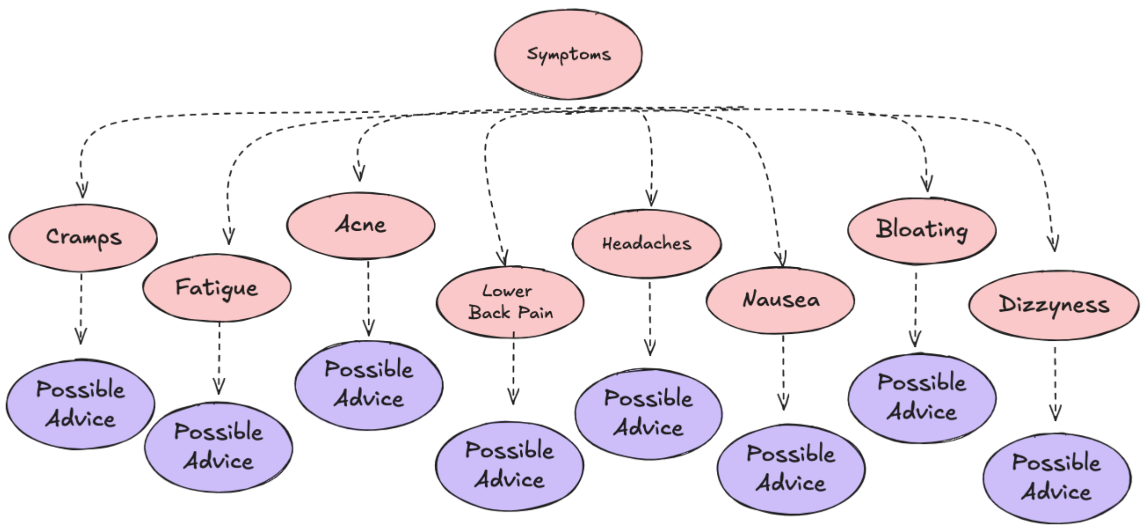
4.3 Data stores:

The data of the period dates is stored under the user account in the database. Based on this information, the displayed dates on the calendar will change.

4.4 Preliminary thoughts on the transition from the logical design to an implementation:

To implement this page, we will need HTML/CSS for the graphic elements as well as Javascript to calculate and display the 6-month calendar. Thus, there will be 2 unique Javascript and HTML files along with a shared CSS file. These coding files will be placed in the development branch with the menu option as its name. This is the only menu option that presents a connection with another menu option, therefore, proper connections for networking will be made.

5. Symptoms module



The symptoms screen appears in a survey format. The user checks the symptoms and depending on the symptoms, it displays possible painkillers that can be used. It does include a warning that these are just recommendations, and the user is responsible for any secondary effects of taking these and that seeing a doctor is likely the best option for them.

This is a one-to-one system where the number of outputs depends on the number of inputs that the user provides, for each input one output is going to be displayed, since there are 8 different inputs, there are 8 different outputs. For now, each output is labeled as possible advice as the team needs to do some research to provide the appropriate advice. The data flow diagram (DFDS) is shown below:

5.1 Process:

Description: This process handles the user's input of symptoms via checkboxes of determined symptoms. It checks which symptoms were marked and displays suggestions for each specific symptom. Detailed descriptions of the process sequencing can be seen in the sequence diagram DFDS

Inputs:

Data (incoming): Symptoms provided via the checkboxes

Outputs:

Data (outgoing): Suggestions provided via text output

5.2 Data flow:

Description: This represents the information being passed from the user regarding their symptoms.

Source: User (external entity)

Destination: Symptoms process

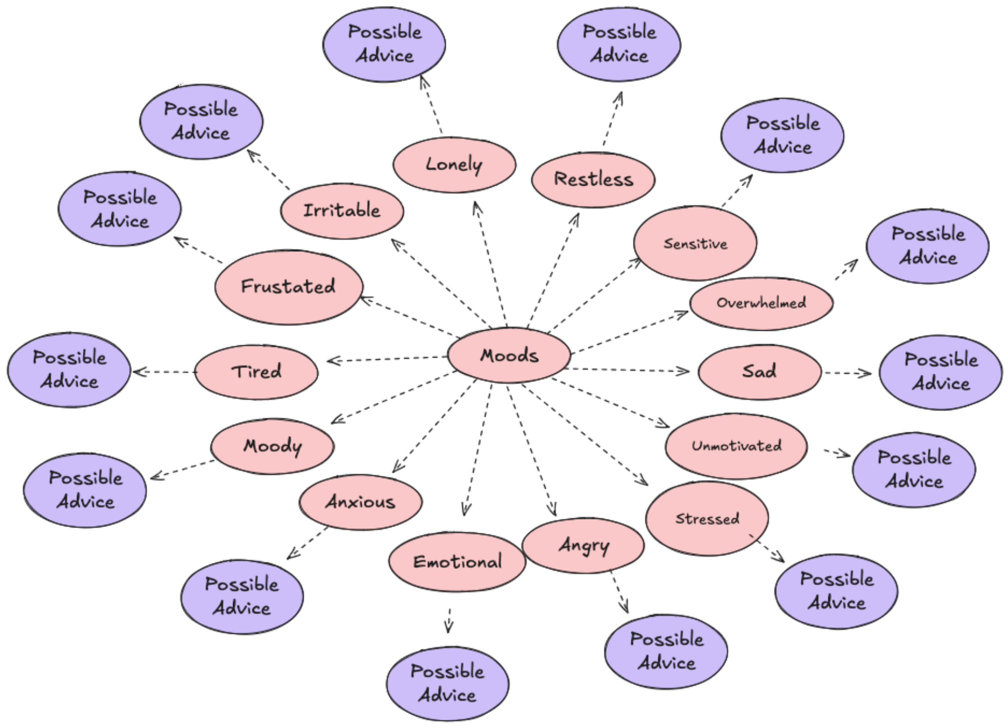
5.3 Data stores:

No data is stored from the input or output, when the user is done using the suggestions, they have the option of clearing and using it once again, or using another menu option from the website.

5.4 Preliminary thoughts on the transition from the logical design to an implementation:

The programming languages that are going to be used are HTML, CSS, and JavaScript. HTML and CSS take care of the graphic elements, and JavaScript will take care that each input displays its proper output, meaning there are going to be three files for this menu option. These coding files will be placed in the development branch with the menu option as its name.

6. **Moods module:**



The Moods page appears in a survey format where the user checks the moods and depending on the moods, it displays possible activities that can be done to improve their mood. Similar to the above Symptoms module, this page is a one-to-one system where the number of outputs depends on the number of inputs that the user provides. For each input, one output is going to be displayed and since there are 14 different inputs, there are 14 different outputs. For now, each output is labeled as possible advice as the team needs to do some research to provide the appropriate advice.

6.1 Process:

Description: This process handles the input of moods of the user via checkboxes of determined moods. It checks which moods were marked and it displays the suggestions for each specific mood. Detailed descriptions of the process sequencing can be seen in the sequence diagram DFDM

Inputs:

Data (incoming): Moods provided via the checkboxes

Outputs:

Data (outgoing): Suggestions provided via text output

6.2 Data flow:

Description: This represents the information being passed from the user regarding their moods.

Source: User (external entity)

Destination: Moods process

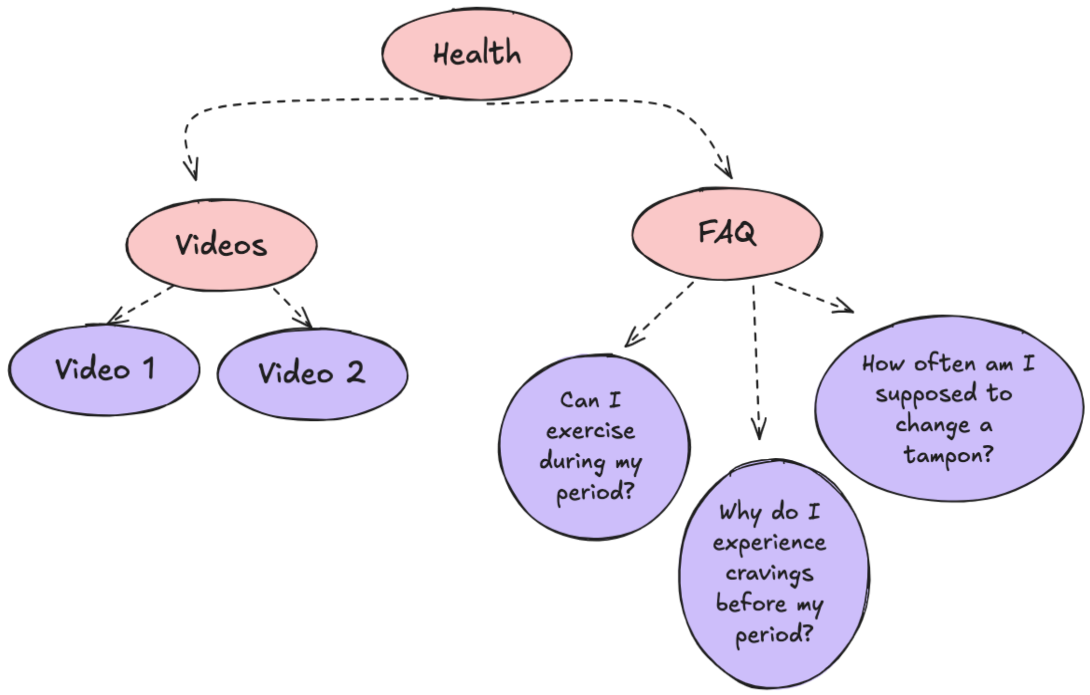
6.3 Data stores:

No data is stored from the input or output. When the user is done using the suggestions, they have the option of clearing it and using it once again or using another menu option from the website.

6.4 Preliminary thoughts on the transition from the logical design to an implementation:

The programming languages that are going to be used are HTML, CSS, and JavaScript. HTML and CSS take care of the graphic elements, and JavaScript will take care that each input displays its proper output, meaning there are going to be three files for this menu option. These coding files will be placed in the development branch with the menu option as its name.

7. **Health module:**


Health
Videos
FAQ
Video 1
Video 2
Can I exercise during my period?
Why do I experience cravings before my period?
How often am I supposed to change a tampon?

The health page provides helpful information through two main options: videos and frequently asked questions (FAQ). Users can choose to watch videos for anyone needing help on any period-related topics. The Users can use the FAQ lists to quickly find answers to common concerns. This page is designed to offer direct and clear information without requiring any input from the user. As the Health page is purely for information delivery, no data flow or storage is needed. The page serves as a static resource where users can access reliable health content related to periods. For more information refer to DFDH.

7.1 Process:

Description: The Health page provides users with menstrual health-related information through videos and FAQ Lists. Users can choose to watch videos or read the FAQs to gain relevant health knowledge, and no traditional data input or output occurs during this process. User interaction is limited to selecting a video to play or an FAQ to view, with no traditional data input or output involved.

Input: No user input

Output: Displaying videos or FAQ content.

7.2 Data flow:

Since the Health page is designed purely to display information in the form of videos and FAQs, no data flow occurs. Users interact by selecting a video or FAQ to view, but no data is transmitted or processed.

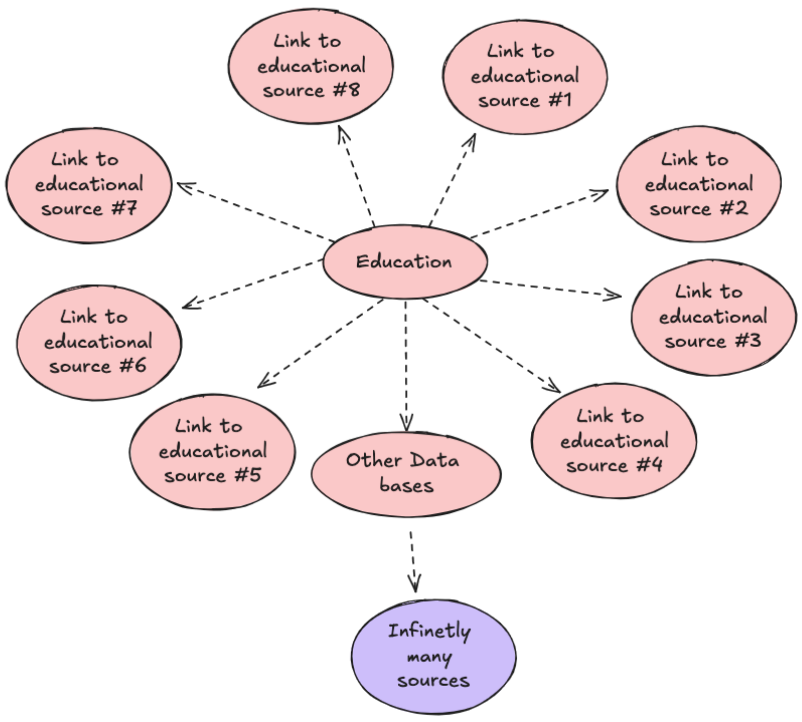
7.3 Data stores:

No data is stored from user interactions. All video and FAQ content is preloaded from the server and displayed directly to the user.

7.4 Preliminary thoughts on the transition from the logical design to an implementation:

The Health module will use HTML and CSS for layout and design. JavaScript can be used to manage the dynamic display of videos and FAQs. Video playback can be implemented through embedded video players, while FAQ content can be displayed using expandable question lists. Since all content is static and preloaded, no backend data processing is required.

8. **Education module:**



Link to educational source #8
Link to educational source #1
Link to educational source #7
Link to educational source #2
Education
Link to educational source #6
Link to educational source #3
Link to educational source #5
Other Data bases
Link to educational source #4
Infinetly many sources

The Education page provides users with external links to health resources. These resources will cover a wide range of topics related to period health, offering more in-depth information for users who wish to explore further in the given topic. Similar to the Health page, the Education page does not require any input from the user. It is designed to guide users to external websites where they can access additional educational materials. Each link corresponds to a specific external source, and these links will open in a new tab, allowing users to easily return to the Education page afterward. This page does not collect or store any user data, as it simply serves as a gateway to more information. For more information, refer to DFDE.

8.1 Process:

When a user clicks a link, it opens in a new tab, allowing the user to view the resource while keeping the original page available.

8.2 Data flow:

The data flow on the Education page is minimal, as it primarily involves the user clicking on a static link that triggers the browser to open the external resource in a new tab. No data is transmitted between the Education page and the external website, and no information is processed or stored by the system during this action.

8.3 Data stores:

The Education module stores the list of external links, which are static and displayed to users. No user data or behaviour is recorded or stored.

8.4 Preliminary thoughts on the transition from the logical design to an implementation:

The Education module can be implemented using HTML and CSS to create a clear list of static links. Each link will be hardcoded into the page, allowing users to click and open the external resources in a new browser tab, and the links are static. This straightforward approach provides users with easy access to external resources without the need for additional complexity in development.

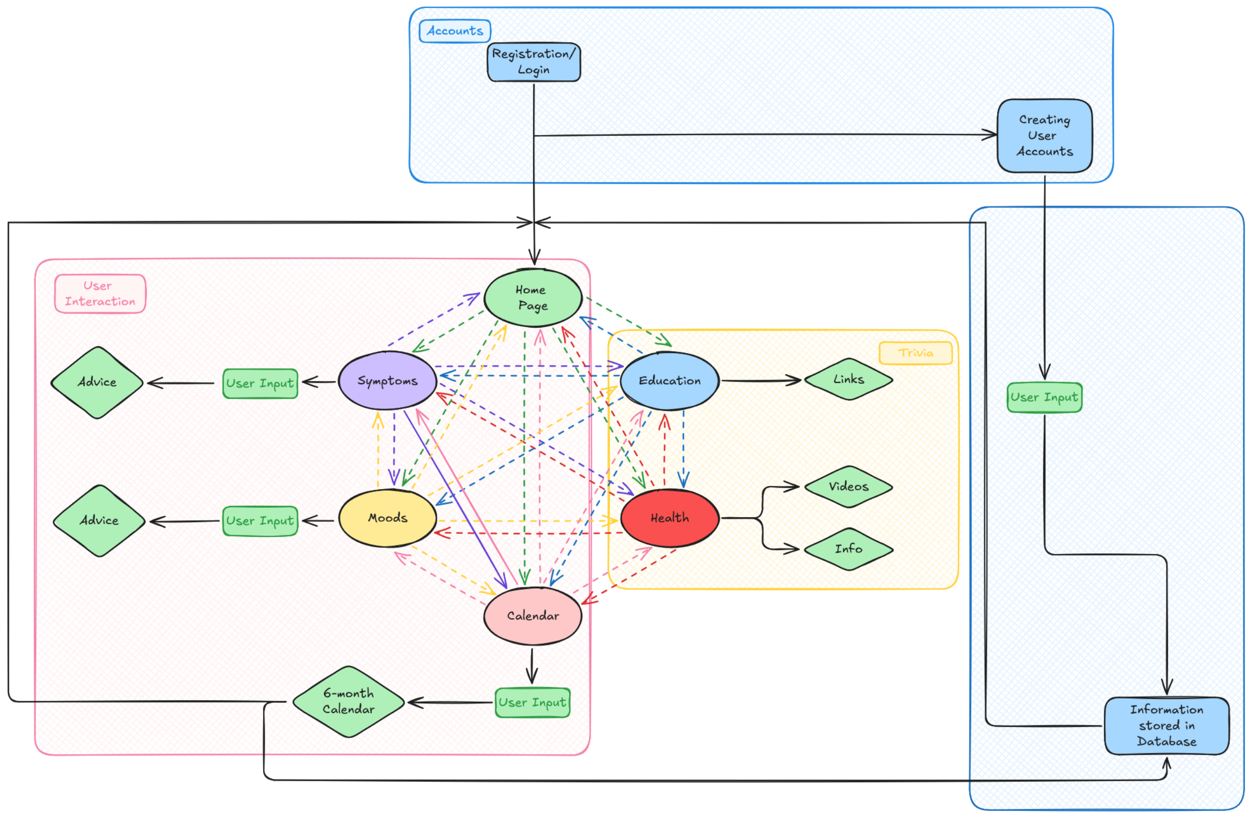
Transition to Physical Design:

Core Decisions:

As briefly discussed in the product overview and throughout the Logical Design section, we plan to implement this project using HTML/CSS for the web page structure, Javascript for the period tracker portion of the website and Javascript/MySQL database for the user accounts. These choices were made as these were languages familiar to some of the members in our group and with all of our busy schedules, sticking to something more familiar was seen as a good idea.

Most of the specific preliminary implementation thoughts have been detailed in each part of the ‘Logical Design’ section above.

Appendix: The Grand Design:



This diagram takes parts from all of the previous diagrams and combines them. Some parts have been simplified for ease of reading, such as all user inputs being simplified to ‘User Input’. Each part is discussed in more detail in the above ‘Logical Design’ section.