Prototype Needs:

* Convert all Figma screens into code
  + Each person is responsible for one screen?
* Determine what is missing from transfer
* Allocate the missing pieces to students
  + Those who have fewer missing pieces/faster at coding
* Build database to link to code
* Continue with usability testing to determine any changes to screens we need to make

Potential tasks to be disbursed:

* User Authentication and Onboarding:
  + Implement user authentication features such as signup, login, and password reset.
  + Develop the onboarding process to guide users through setting up their profiles and preferences.
  + Figma does not have much of this- figure out how to connect a user database to the UX/UI
  + Build user database table
* Dashboard and Overview:
  + Create the main dashboard interface
  + Display key metrics and insights prominently for easy access.
  + Build key metrics table
* Cycle Tracking and Logging:
  + Implement features for users to input and track their hormone cycle data.
  + Allow users to log various parameters such as menstrual flow, mood, sleep, etc.
  + Build parameters table
* Calendar View and Cycle Prediction:
  + Develop a calendar view to visualize the user's hormone cycle over time.
  + Implement algorithms to predict future cycle phases based on historical data?
  + What database needs do we have for this?
    - What Chat says about this:
      * Install Cordova Plugins:
        + For Android:

Use the Cordova Local Notification plugin. Install it using the following command:

Sql

cordova plugin add cordova-plugin-local-notification

* + - * + For iPhone (iOS):

Use the Cordova Push Notifications plugin. Install it using the following command:

Csharp

cordova plugin add phonegap-plugin-push

* + - * Configure Platform-Specific Settings:
        + For Android:

Make sure your AndroidManifest.xml is configured properly to handle notifications. Check the plugin documentation for any additional configuration steps.

* + - * + For iPhone (iOS):

You'll need to set up push notification certificates in your Apple Developer account and configure your Xcode project accordingly. Follow the instructions provided by the plugin documentation and Apple's guidelines.

* + - * Implement Notification Logic:
        + In your JavaScript code (typically in your app's main script file), implement the logic for triggering and handling notifications based on your app's requirements.
      * Use the appropriate methods provided by the Cordova plugin(s) to schedule and display notifications.
      * Test Your Notifications:
        + Test your notifications on both Android and iOS devices or emulators to ensure they work as expected.
        + For iOS, you may need to deploy your app to a physical device to test notifications due to restrictions on push notifications in iOS simulators.
      * Handle Background and Foreground Notifications (Optional):
        + Depending on your app's requirements, you may need to handle notifications differently when the app is in the background or foreground. Implement appropriate logic to handle these scenarios.
      * Handle Notification Data (Optional)
* Reminder and Notification System:
  + Build a system for setting reminders for important events related to the hormone cycle.
  + Implement push notifications to remind users of upcoming cycle phases or tasks.(see above)
* Data Analysis and Insights:
  + Develop features to analyze the user's hormone cycle data and provide insights.
  + Implement charts, graphs, or reports to visualize trends and patterns in the data.
  + Not sure how to do this in code- chat says:
    - Choose a Charting Library:
    - There are several charting libraries available for JavaScript that you can use in your mobile app. Some popular options include Chart.js, D3.js, and Highcharts. Choose one that fits your project requirements and has good compatibility with mobile platforms.
    - Install the Charting Library:
    - If you haven't already installed the charting library, you can do so using npm (Node Package Manager). Open a terminal in VS Code and run the command to install the library. For example:
    - Copy code
    - npm install chart.js
    - Replace chart.js with the name of the charting library you've chosen.
    - Include the Library in Your Project:
    - Once the library is installed, include it in your HTML file by adding a <script> tag with the source pointing to the library file. For example:
    - Html:
      * <script src="node\_modules/chart.js/dist/Chart.min.js"></script>
    - Create a Canvas Element:
    - In your HTML file, create a <canvas> element where the chart will be rendered. Give it an ID to reference it in your JavaScript code. For example:
    - Html:
      * <canvas id="myChart"></canvas>
    - Write JavaScript Code to Generate the Chart:
    - In your JavaScript file, write code to generate the chart using the charting library. This typically involves creating a new instance of the chart object and passing configuration options and data. For example, using Chart.js:
    - javascript:
    - var ctx = document.getElementById('myChart').getContext('2d');

var myChart = new Chart(ctx, {

type: 'bar',

data: {

labels: ['Red', 'Blue', 'Yellow', 'Green', 'Purple', 'Orange'],

datasets: [{

label: '# of Votes',

data: [12, 19, 3, 5, 2, 3],

backgroundColor: [

'rgba(255, 99, 132, 0.2)',

'rgba(54, 162, 235, 0.2)',

'rgba(255, 206, 86, 0.2)',

'rgba(75, 192, 192, 0.2)',

'rgba(153, 102, 255, 0.2)',

'rgba(255, 159, 64, 0.2)'

],

borderColor: [

'rgba(255, 99, 132, 1)',

'rgba(54, 162, 235, 1)',

'rgba(255, 206, 86, 1)',

'rgba(75, 192, 192, 1)',

'rgba(153, 102, 255, 1)',

'rgba(255, 159, 64, 1)'

],

borderWidth: 1

}]

},

options: {

scales: {

y: {

beginAtZero: true

}

}

}

});

* + - Test Your Chart:
      * Test your chart in both Android and iPhone devices or emulators to ensure it displays correctly and behaves as expected on both platforms.
    - Integrate the Chart into Your Mobile App:
      * Once your chart is working as desired, integrate it into your mobile app along with other components and functionalities.
* Settings and Preferences:
  + Create a settings interface for users to customize their app experience.
  + Allow users to adjust preferences such as notification settings, units of measurement, etc.
* Testing and Quality Assurance
  + Set up comprehensive testing procedures to ensure the app functions correctly across different devices and scenarios.
  + Perform both manual and automated testing to identify and fix any bugs or issues.