HerHealth

A PERSONALIZED CYCLE TRACKING AND PREDICTION SYSTEM

Ankita Muni | PES1UG23CS081
Likitha H | PES1UG24CS810

Abstract

The **HerHealth** system is a web-based application designed to assist users in tracking their menstrual cycles, managing associated health metrics, and receiving future cycle predictions and medication reminders. The core problem addressed is the need for a comprehensive, personalized, and proactive health tracking tool. The system allows users to log cycle start/end dates, mood swings, weight, and height, as well as upload doctor consultation files for medication records. The application utilizes historical cycle data to calculate and predict the next cycle and ovulation dates, automatically generating pending notifications to ensure users are prepared for their upcoming cycle and have adequate medication stock.

User Requirement Specification

ID	**Requirement Type**	**Description**
UR-001	User Management	The system shall allow new users to **register** by providing personal details (name, email, phone, DOB, location) and a password.
UR-002	User Management	The system shall allow registered users to **log in** using their email and password.
UR-003	Cycle Logging (CRUD)	Users shall be able to **log a new cycle** by providing the start date, end date, mood swings, weight, and height.
UR-004	Cycle Logging (CRUD)	The system shall automatically calculate the **cycle length** and update the database upon logging a new cycle.
UR-005	Prediction	The system shall calculate the **next predicted cycle start/end** and **ovulation date** based on the average of the last 10 full cycle lengths.
UR-006	Notification	The system shall automatically **generate a notification** for the next predicted cycle, including a medication stock reminder.
UR-007	Medication Tracking (CRUD)	Users shall be able to **log medicine** details (name, dosage) for their current cycle.

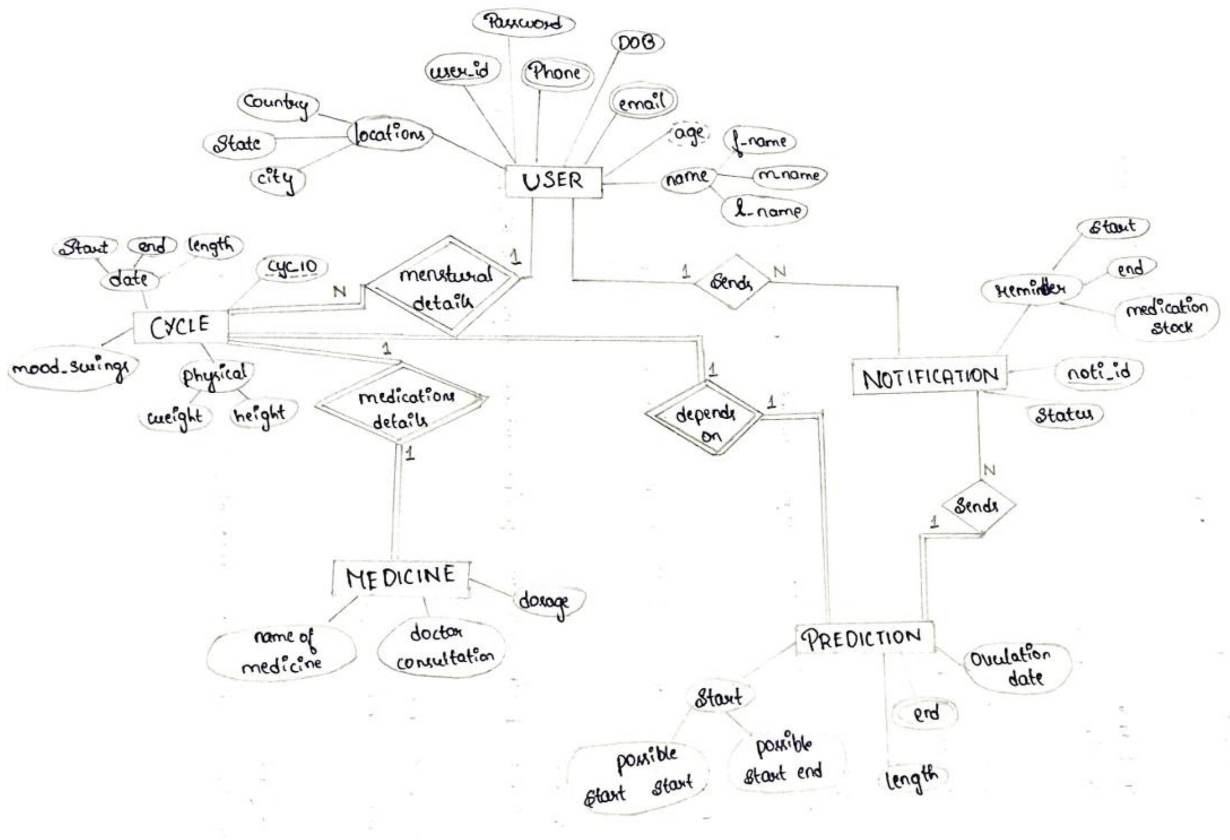
ID	Requirement Type	Description
UR-008	Consultation Upload	Users shall be able to **upload a doctor consultation file** (as a BLOB) when logging medicine.
UR-009	Data Retrieval	Users shall be able to view a **history** of all past cycles and associated medications.
UR-010	Notification Management	Users shall be able to **dismiss/complete** a pending notification.
UR-011	Data Presentation	The dashboard shall display **pending notifications**, the **latest prediction**, and a **chart** of historical data (period length, weight, cycle length).
UR-012	Administration	The system shall include logic for **archiving completed notifications** older than a specified duration (Procedure).

List of Software/Tools/Programming Languages Used

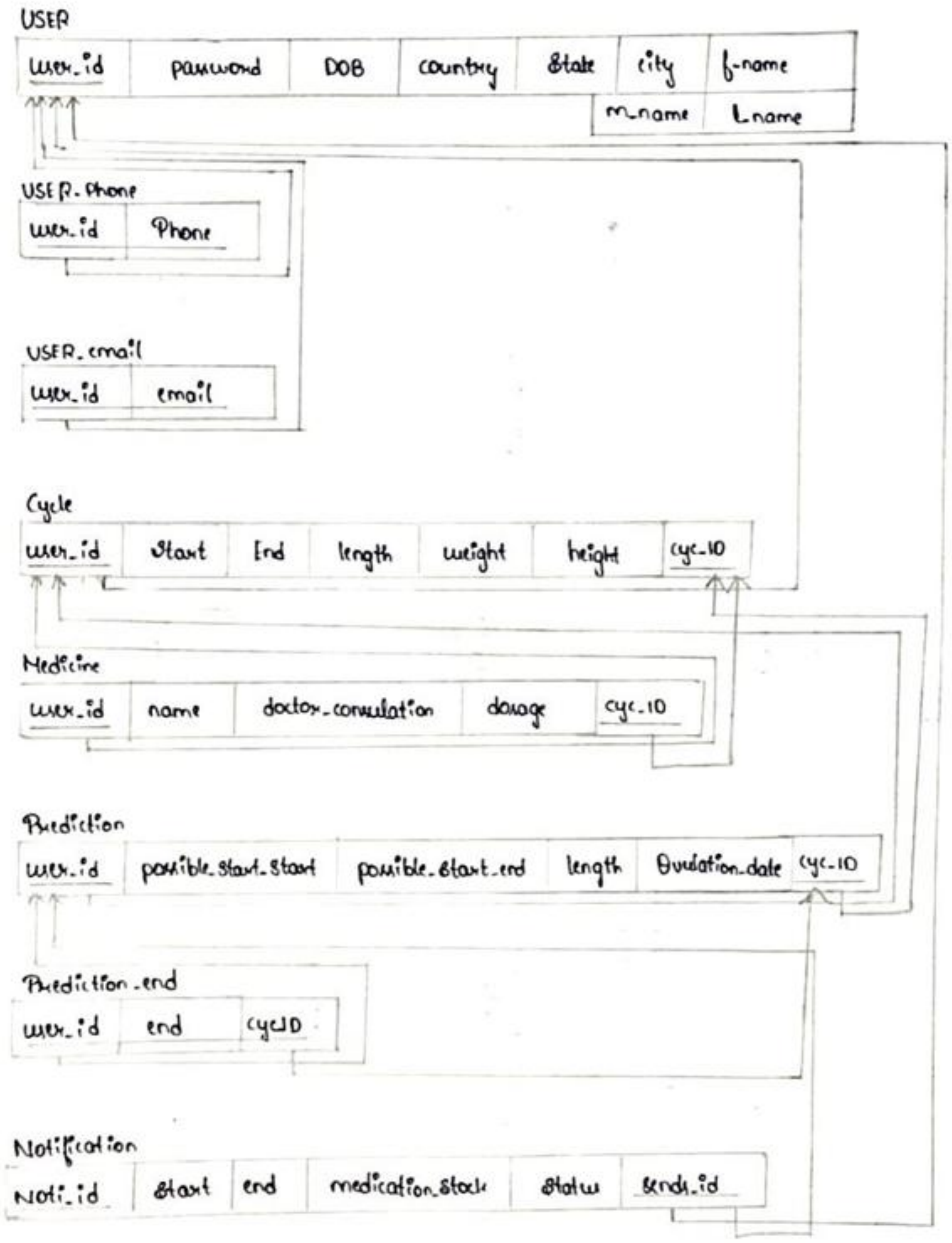
Category	Tool / Language	Specific Version / Library
Programming Language	Python	3.x
Database Management	MySQL	
Web Framework	Flask	[1]
Database Connector	mysql-connector-python	[2]

Category	Tool / Language	Specific Version / Library
Security	bcrypt	For password hashing [3]
Front-end	HTML, CSS, JavaScript	(Assumed for web interface)

ER Diagram



Relational Schema


USER
user_id
password
DOB
country
State
city
f-name
m_name
L_name
USER-Phone
user_id
Phone
USER_email
user_id
email
Cycle
user_id
Start
End
length
weight
height
cyc-ID
Medicine
user_id
name
doctor_consulation
dosage
cyc-ID
Prediction
user_id
possible_start_start
possible_start_end
length
Ovulation_date
cyc-ID
Prediction_end
user_id
end
cycID
Notification
Noti_id
Start
end
medication_stock
Status
sender_id

DDL Commands (Data Definition Language)

1. Database Creation

-- Creates the database

CREATE DATABASE herhealth;

-- Selects the database for use

USE herhealth;

---

2. Table and Constraint Definitions (CREATE TABLE)

These commands establish the five main entities and the archive table, defining primary keys (PK), foreign keys (FK), unique constraints, and check constraints.

Table	**Command Snippet (Key DDL Elements)**

user	create table user (
	    user_id int auto_increment primary key,
	    f_name varchar(50),
	    m_name varchar(50),
	    l_name varchar(50) not null,
	    email varchar(100) unique not null,
	    phone varchar(15) not null,
	    dob date,
	    password varchar(100) not null,
	    country varchar(50) default 'india',
	    state varchar(50),
	    city varchar(50),
	    constraint chk_email check (email like '%@%')
	);

cycle	create table cycle (

Table	Command Snippet (Key DDL Elements)
	cycle_id int auto_increment primary key, user_id int, start_date date not null, end_date date, length int check (length > 0), mood_swings enum('none', 'mild', 'moderate', 'severe') default 'none', weight decimal(5,2) check (weight > 0), height decimal(5,2) check (height > 0), foreign key (user_id) references user(user_id) on delete cascade);
medicine	create table medicine (med_id int auto_increment primary key, cycle_id int, name_of_medicine varchar(100) not null, dosage varchar(50), doctor_consultation mediumblob, foreign key (cycle_id) references cycle(cycle_id) on delete cascade);
notification	s create table notification (noti_id int auto_increment primary key, user_id int, start_date date, end_date date, medication_stock varchar(100), status enum('pending', 'sent', 'completed') default 'pending', foreign key (user_id) references user(user_id) on delete cascade

Table	Command Snippet (Key DDL Elements)
	);

prediction	create table prediction (
	    prediction_id int auto_increment primary key,
	    cycle_id int,
	    noti_id int,
	    possible_start_start date,
	    possible_start_end date,
	    ovulation_date date,
	    end date,
	    length int check (length > 0),
	    foreign key (cycle_id) references cycle(cycle_id) on delete cascade,
	    foreign key (noti_id) references notification(noti_id) on delete cascade
	);

notification_archive	CREATE TABLE `notification_archive` (
	  `noti_id` int NOT NULL,
	  `user_id` int DEFAULT NULL,
	  `start_date` date DEFAULT NULL,
	  `end_date` date DEFAULT NULL,
	  `medication_stock` varchar(100) DEFAULT NULL,
	  `status` varchar(10) DEFAULT 'archived',
	  PRIMARY KEY (`noti_id`)
	);

---

3. Table Modifications (ALTER TABLE)

-- Adds an enumeration column to distinguish user roles

ALTER TABLE user

```
ADD COLUMN role ENUM('user', 'admin') NOT NULL DEFAULT 'user';


-- Adds columns to handle file details for the BLOB data in medicine
ALTER TABLE medicine
ADD COLUMN consultation_filename VARCHAR(255) NULL,
ADD COLUMN consultation_mimetype VARCHAR(100) NULL;
```

---

4. Stored Program Definitions (Functions, Procedures, Triggers)

A. Function (CREATE FUNCTION)

SQL

```
DELIMITER $$
CREATE FUNCTION get_latest_medication_dosage (
    p_user_id INT,
    p_med_name VARCHAR(100)
)
RETURNS VARCHAR(50)
READS SQL DATA
BEGIN
    DECLARE latest_dosage VARCHAR(50);

    SELECT m.dosage INTO latest_dosage
    FROM medicine m
    JOIN cycle c ON m.cycle_id = c.cycle_id
    WHERE c.user_id = p_user_id
      AND m.name_of_medicine = p_med_name
    ORDER BY c.start_date DESC
    LIMIT 1;

    RETURN latest_dosage;
```

```
END$$
DELIMITER ;
```

B. Procedure (CREATE PROCEDURE)

SQL

```
DELIMITER $$
DROP PROCEDURE IF EXISTS `archive_completed_notifications`$$ -- DDL to drop if exists

CREATE PROCEDURE `archive_completed_notifications` (
  IN p_days_old INT
)
BEGIN
  DECLARE v_cutoff_date DATE;
  SET v_cutoff_date = DATE_SUB(CURDATE(), INTERVAL p_days_old DAY);

  -- ... Transaction logic (DML within DDL definition) ...

END$$
DELIMITER ;
```

C. Trigger (CREATE TRIGGER)

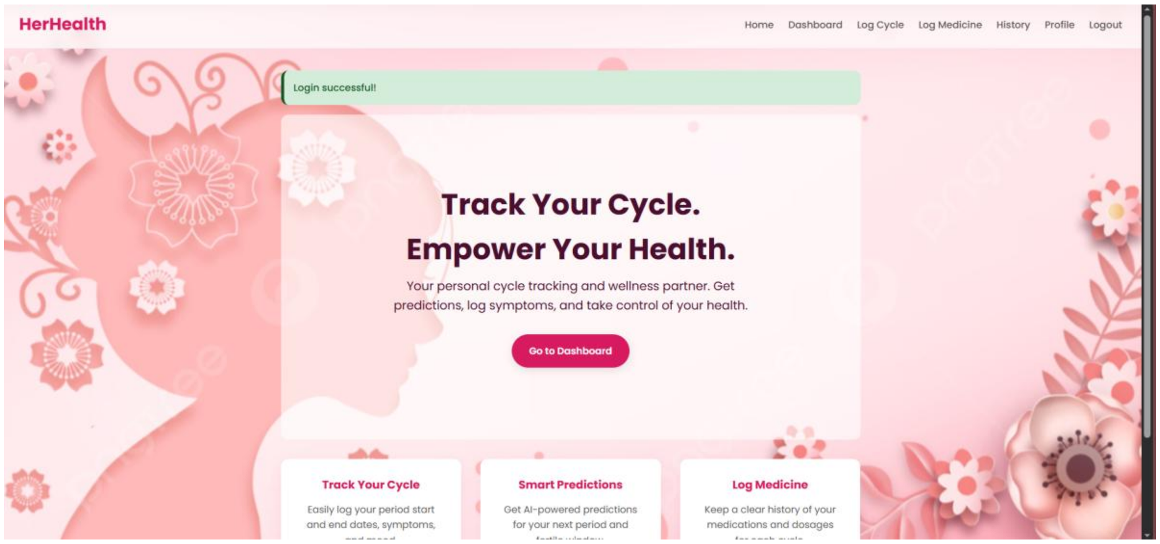
SQL

```
DELIMITER $$
CREATE TRIGGER before_insert_prediction_dates
BEFORE INSERT ON prediction
FOR EACH ROW
BEGIN
  DECLARE notification_start_date DATE;
  DECLARE notification_end_date DATE;

  SELECT start_date, end_date
```

```
    INTO notification_start_date, notification_end_date
    FROM notification
    WHERE noti_id = NEW.noti_id;

    IF NEW.possible_start_start IS NULL THEN
        SET NEW.possible_start_start = notification_start_date;
    END IF;

    IF NEW.end IS NULL THEN
        SET NEW.END = notification_end_date;
    END IF;

    IF notification_start_date IS NOT NULL AND notification_end_date IS NOT NULL THEN
        SET NEW.length = DATEDIFF(notification_end_date, notification_start_date) + 1;
    END IF;
END$$
DELIMITER ;
```

CRUD operation Screenshots

CREATE



HerHealth

Home Dashboard Log Cycle Log Medicine History Profile Logout

Login successful!

Track Your Cycle.
Empower Your Health.

Your personal cycle tracking and wellness partner. Get predictions, log symptoms, and take control of your health.

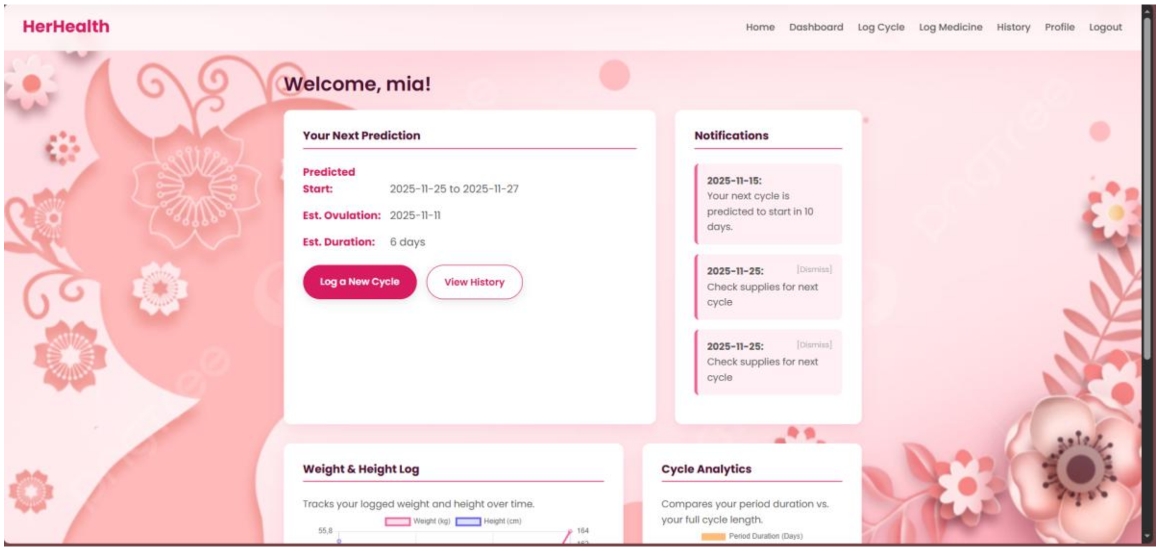
Go to Dashboard

Track Your Cycle
Easily log your period start and end dates, symptoms,

Smart Predictions
Get AI-powered predictions for your next period and

Log Medicine
Keep a clear history of your medications and dosages

READ



HerHealth

Home Dashboard Log Cycle Log Medicine History Profile Logout

Welcome, mia!

Your Next Prediction

Predicted Start:	2025-11-25 to 2025-11-27
Est. Ovulation:	2025-11-11
Est. Duration:	6 days

Log a New Cycle View History

Notifications

2025-11-15: Your next cycle is predicted to start in 10 days.

2025-11-25: [Dismiss] Check supplies for next cycle

2025-11-25: [Dismiss] Check supplies for next cycle

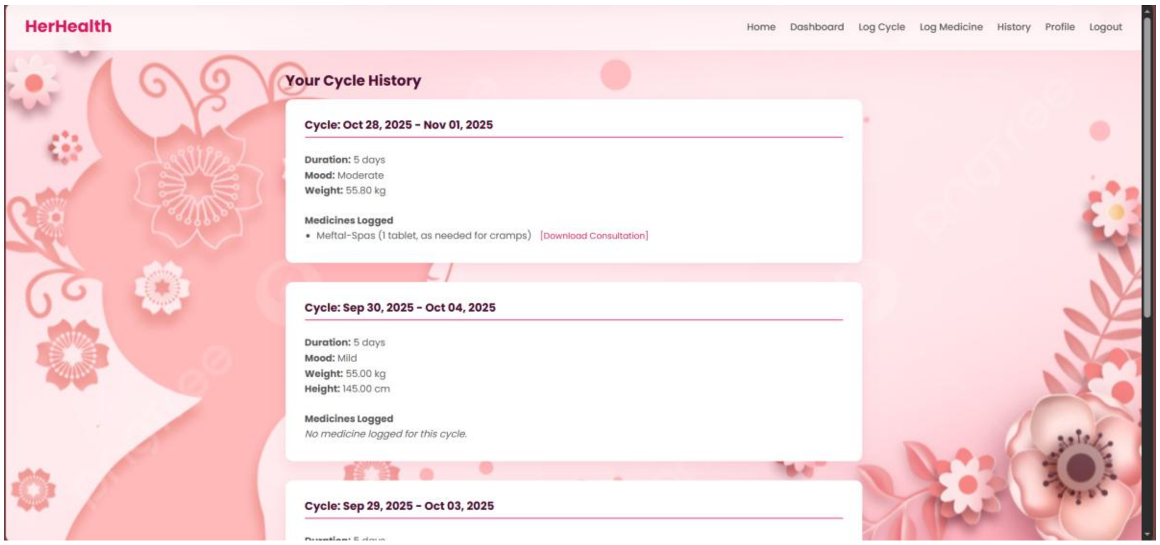
Weight & Height Log

Tracks your logged weight and height over time.

Cycle Analytics

Compares your period duration vs. your full cycle length.

UPDATE


HerHealth
Home Dashboard Log Cycle Log Medicine History Profile Logout

Your Cycle History

Cycle: Oct 28, 2025 - Nov 01, 2025

Duration: 5 days
Mood: Moderate
Weight: 55.80 kg

Medicines Logged
- Meftal-Spas (1 tablet, as needed for cramps) [Download Consultation]

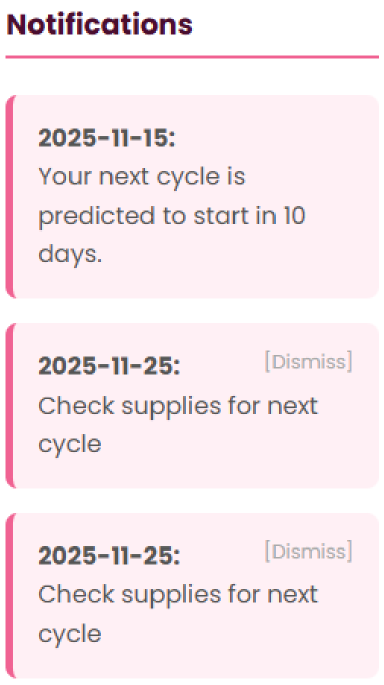
Cycle: Sep 30, 2025 - Oct 04, 2025

Duration: 5 days
Mood: Mild
Weight: 55.00 kg
Height: 145.00 cm

Medicines Logged
No medicine logged for this cycle.

Cycle: Sep 29, 2025 - Oct 03, 2025

DELETE



Notifications

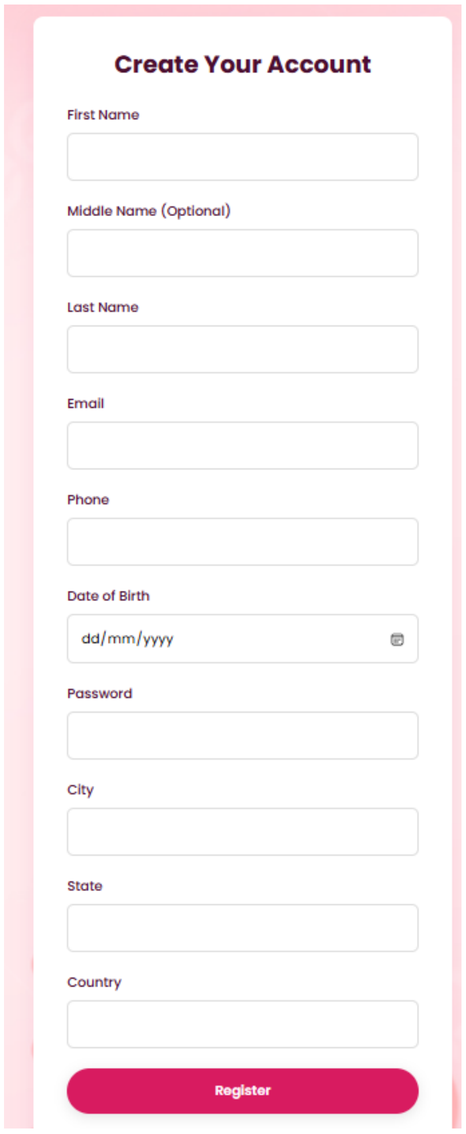
2025-11-15: Your next cycle is predicted to start in 10 days.

2025-11-25: [Dismiss]
Check supplies for next cycle

2025-11-25: [Dismiss]
Check supplies for next cycle

List of functionalities/features of the application and its associated screenshots using front end

1. User Authentication



Create Your Account

First Name

Middle Name (Optional)

Last Name

Email

Phone

Date of Birth

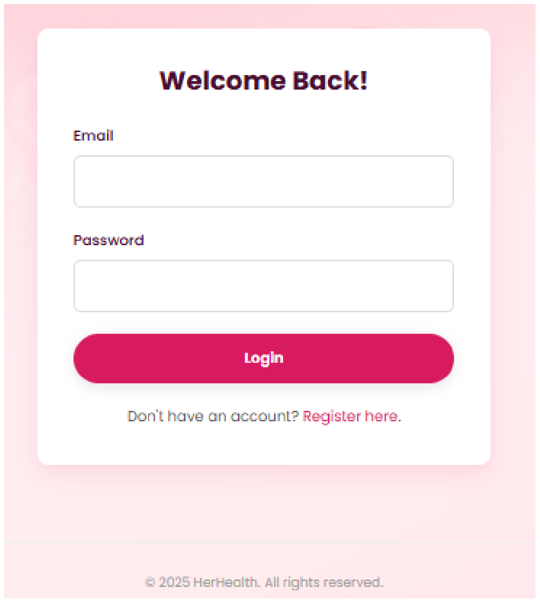
dd/mm/yyyy

Password

City

State

Country



Welcome Back!

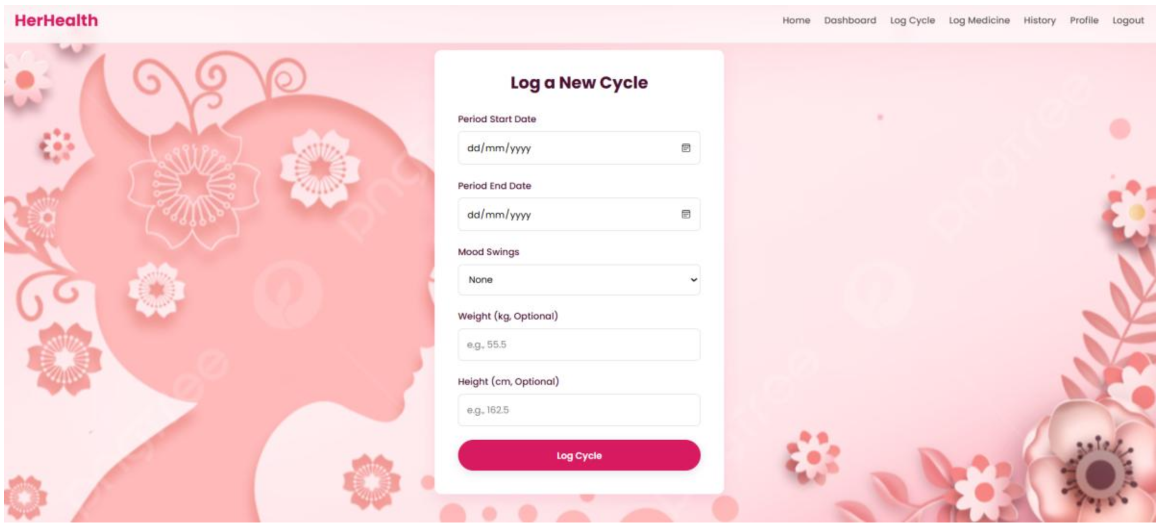
Email

Password

Don't have an account? Register here.

© 2025 HerHealth. All rights reserved.

2. Cycle Logging



Log a New Cycle

Period Start Date: dd/mm/yyyy

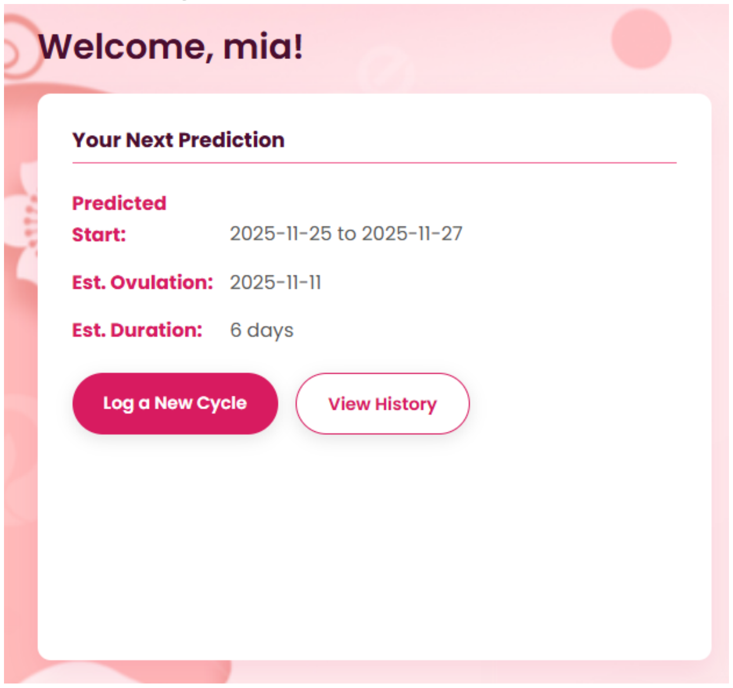
Period End Date: dd/mm/yyyy

Mood Swings: None

Weight (kg, Optional): e.g., 55.5

Height (cm, Optional): e.g., 162.5

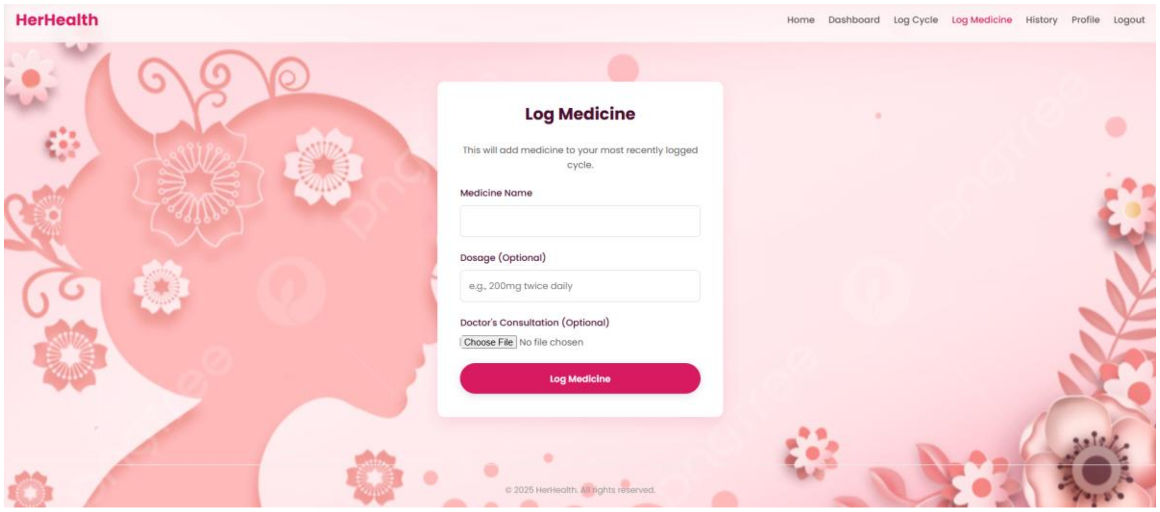
3. Prediction Engine


Welcome, mia!

Your Next Prediction

Predicted Start:	2025-11-25 to 2025-11-27
Est. Ovulation:	2025-11-11
Est. Duration:	6 days

4. Medication Tracking



Log Medicine

This will add medicine to your most recently logged cycle.

Medicine Name

Dosage (Optional)

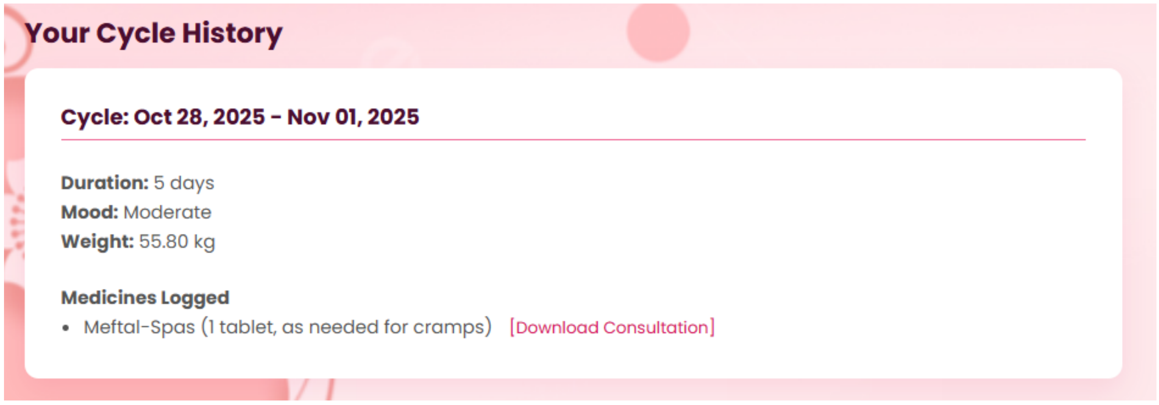
e.g., 200mg twice daily

Doctor's Consultation (Optional)

Choose File No file chosen

© 2025 HerHealth. All rights reserved.

5. File Upload/Download



Your Cycle History

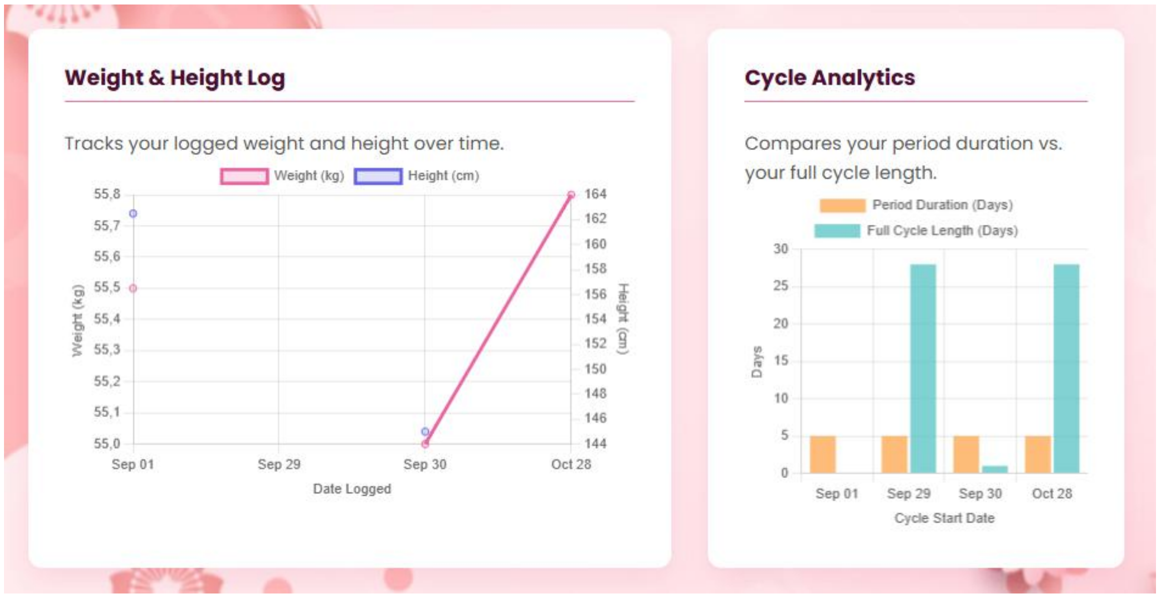
Cycle: Oct 28, 2025 - Nov 01, 2025

Duration: 5 days
Mood: Moderate
Weight: 55.80 kg

Medicines Logged

- Meftal-Spas (1 tablet, as needed for cramps) [Download Consultation]

6. Data Visualization



Weight & Height Log

Tracks your logged weight and height over time.

Cycle Analytics

Compares your period duration vs. your full cycle length.

Triggers, Procedures/Functions, Nested query, Join, Aggregate queries

Type	**Name / Purpose**	**Code Reference**

Trigger	before_insert_prediction_dates	delimiter $$
		-- Sets prediction dates
		create trigger before_insert_prediction_dates
		before insert on prediction
		for each row
		begin
		    declare notification_start_date date;
		    declare notification_end_date date;
		
		    select start_date, end_date
		    into notification_start_date, notification_end_date
		    from notification
		    where noti_id = new.noti_id;
		
		    if new.possible_start_start is null then
		        set new.possible_start_start = notification_start_date;
		    end if;
		
		    if new.end is null then
		        set new.end = notification_end_date;
		    end if;
		
		    if notification_start_date is not null and notification_end_date is not null then

Type	Name / Purpose	Code Reference

		set new.length = datediff(notification_end_date, notification_start_date) + 1;
		end if;
		end$$
		delimiter ;

Procedure	archive_completed_notifications	DELIMITER $$ -- Drops procedure if exists DROP PROCEDURE IF EXISTS \`archive_completed_notifications\`; -- Archives old notifications CREATE PROCEDURE \`archive_completed_notifications\` (IN p_days_old INT) BEGIN DECLARE v_cutoff_date DATE; SET v_cutoff_date = DATE_SUB(CURDATE(), INTERVAL p_days_old DAY); -- Start transaction START TRANSACTION; -- Copy to archive INSERT INTO notification_archive (noti_id, user_id, start_date, end_date, medication_stock,

Type	Name / Purpose	Code Reference

		`status`
		`)`
		`SELECT`
		`noti_id,`
		`user_id,`
		`start_date,`
		`end_date,`
		`medication_stock,`
		`'archived'`
		`FROM notification`
		``WHERE `status` = 'completed'``
		``AND `start_date` < v_cutoff_date;``
		`-- Delete from original`
		`DELETE FROM notification`
		``WHERE `status` = 'completed'``
		``AND `start_date` < v_cutoff_date;``
		`-- Commit changes`
		`COMMIT;`
		`-- Show rows affected`
		`SELECT ROW_COUNT() AS 'archived_count';`
		`END$$`
		`DELIMITER ;`

Function	get_latest_medication_dosage	`delimiter $$`
		`-- Gets latest medication dosage`
		`create function get_latest_medication_dosage (`

Type	Name / Purpose	Code Reference

		`p_user_id int,`
		`p_med_name varchar(100)`
		`)`
		`returns varchar(50)`
		`reads sql data`
		`begin`
		`declare latest_dosage varchar(50);`
		`select m.dosage into latest_dosage`
		`from medicine m`
		`join cycle c on m.cycle_id = c.cycle_id`
		`where c.user_id = p_user_id`
		`and m.name_of_medicine = p_med_name`
		`order by c.start_date desc`
		`limit 1;`
		`return latest_dosage;`
		`end$$`
		`delimiter ;`

Nested Query	Find **longest cycle** per user. Uses a Common Table Expression (WITH cycle_rank AS (...)) and a window function (RANK() OVER (...)).	`with cycle_rank as (`
		`select user_id, cycle_id, length,`
		`rank() over (partition by user_id order by length desc) as rnk`
		`from cycle`
		`)`
		`select u.f_name, u.l_name, c.length`
		`from user u`

Type	Name / Purpose	Code Reference
		join cycle_rank c on u.user_id = c.user_id where c.rnk = 1;

JOIN	Find **medications for severe mood swings**. Joins cycle and medicine tables on cycle_id to link health events to prescriptions.	select
		    c.cycle_id,
		    c.start_date,
		    c.mood_swings,
		    m.name_of_medicine,
		    m.dosage
		from
		    cycle c
		join
		    medicine m on c.cycle_id = m.cycle_id
		where
		    c.mood_swings = 'severe'
		order by
		    c.start_date desc;

Aggregate Query	Find user with **highest average BMI**. Uses aggregate functions AVG() and ROUND(), along with GROUP BY and ORDER BY.	select
		    u.user_id,
		    concat(u.f_name, ' ', u.l_name) as full_name,
		    u.city,
		    round(avg(c.weight / (c.height / 100.0 * c.height / 100.0)), 2) as average_bmi
		from
		    user u

Type	Name / Purpose	Code Reference
		join cycle c on u.user_id = c.user_id group by u.user_id, u.f_name, u.l_name, u.city order by average_bmi DESC limit 1;

Code snippets for invoking the Procedures/Functions/Trigger

```
-- Function
delimiter $$
create function get_latest_medication_dosage (
   p_user_id int,
   p_med_name varchar(100)
)
returns varchar(50)
reads sql data
begin
   declare latest_dosage varchar(50);

   select m.dosage into latest_dosage
   from medicine m
   join cycle c on m.cycle_id = c.cycle_id
   where c.user_id = p_user_id
     and m.name_of_medicine = p_med_name
   order by c.start_date desc
```

```
    limit 1;

    return latest_dosage;
end$$

delimiter ;



--  PROCEDURE DEFINITIONS
DELIMITER $$
DROP PROCEDURE IF EXISTS `archive_completed_notifications`;

-- Archives old notifications
CREATE PROCEDURE `archive_completed_notifications` (
    IN p_days_old INT
)
BEGIN
    DECLARE v_cutoff_date DATE;
    SET v_cutoff_date = DATE_SUB(CURDATE(), INTERVAL p_days_old DAY);
    START TRANSACTION;
    INSERT INTO notification_archive (
        noti_id,
        user_id,
        start_date,
        end_date,
        medication_stock,
        status
    )
    SELECT
        noti_id,
        user_id,
        start_date,
        end_date,
        medication_stock,
        'archived'
    FROM notification
    WHERE `status` = 'completed'
      AND `start_date` < v_cutoff_date;
    DELETE FROM notification
    WHERE `status` = 'completed'
      AND `start_date` < v_cutoff_date;
    COMMIT;
    SELECT ROW_COUNT() AS 'archived_count';
END$$
DELIMITER ;

--  TRIGGER DEFINITIONS


delimiter $$
create trigger before_insert_prediction_dates
before insert on prediction
for each row
begin
    declare notification_start_date date;
    declare notification_end_date date;

    select start_date, end_date
    into notification_start_date, notification_end_date
    from notification
    where noti_id = new.noti_id;

    if new.possible_start_start is null then
        set new.possible_start_start = notification_start_date;
    end if;

    if new.end is null then
        set new.end = notification_end_date;
    end if;

    if notification_start_date is not null and notification_end_date is not null then
        set new.length = datediff(notification_end_date, notification_start_date) + 1;
    end if;
end$$
delimiter ;
```

SQL queries(Create, Insert, Triggers, Procedures/Functions, Nested query, Join, Aggregate queries) used in the project in the form of .sql file

HerHealth/herhealth.sql at main · AnkitaMuni/HerHealth

Github repo link

AnkitaMuni/HerHealth