P4:

Question: Where in the source code can you find these energy patterns? Please describe your approach to searching each pattern.

I assume MVC, Model, View, and controller.

4.1 Dark UI Colors

UI component, CSS

4.2 Dynamic Retry Delay

controller where HTTP request are created

4.3 Avoid Extraneous Work

hard to pinpoint, this could be anywhere.

4.4 Race-to-idle

depends on which resource you use and open it, and you will look where it resides.

4.5 Open Only When Necessary

Controller. depends on which resource and where the object reside

4.6 Push over Poll

Just the controller part, front and backend.

4.7 Power Save Mode

It is application dependent. Loading images vs not loading them. depends on the client feature.

It is super hard to answer.

4.8 Power Awareness

it is definitely a client feature, where you load images or where you control web APIs e.g., manage bluetooth.So it can be in view and controller.

4.9 Reduce Size

that is the server side, what proxy are you entering. Server middleware, entering proxy.

4.10 WiFi over Cellular

that is not possible with a desktop computer. Device api calls

4.11 Suppress Logs

This can be everywhere in the source code. I would search for logging libraries and logging commands,

4.12 Batch Operations

Controller side and the client side. If you talk about databases, then the persistence layer of database calls.

4.13 Cache

Look for local storage calls that can be anywhere in the code. The variables stored in the memory. In the model part of MVC, it is where the data reside the most.

4.14 Decrease Rates

controller side, where I have some timeout functions. On the client side.

4.15 User Knows Best

view (MVC) for the users and then will search for their features if users can enable or disable.

4.16 Inform Users

view (MVC) for the users and then will search for their features if they can enable or disable.

4.17 Enough Resolution

Controller in sense, where the data leaves your device.

4.17 Sensor Fusion

device API calls

4.19 Kill abnormal Tasks

On the client-side, I will see if there are timeouts, I will go into the network part. I will just go into developer mode in the browser and go to the network tab and search for long lasting calls.

4.20 No Screen Interaction

device API calls, and controller to your hardware.

4.21 Avoid Extraneous Graphics and Animations

In the view component, If we talk about rendering graphics and view parts like CSS. there are essential libraries, eg CSS, SVG library, UI. will look for a button in the view part.

4.22 Manual Sync, On Demand

Looking for the buttons In the view component, network.
It is tricky to implement it. It touches the view, network, controller.

P5: **Question: Where in the source code can you find these energy patterns? Please describe your approach to searching each pattern.**

4.1 Dark UI Colors

root element of the application, when the root element/file is rendered, e.g. app.js

4.2 Dynamic Retry Delay

All instances, load from server, file to perform API requests.

4.3 Avoid Extraneous Work

All instances where some sort of page rendering mechanism to track the user position on the website and then figure out which tasks are relevant and which are not.

4.4 Race-to-idle

All instances where I communicate with external resources, for example for react and view we have separate files, services files.

4.5 Open Only When Necessary

In all instances where I communicate with external resources, for example for react and view we have a separate file, services files.

4.6 Push over Poll

everywhere i am in communication with server, I will check if I have push method rather than pull

4.7 Power Save Mode

I will find them in sort of root files of the application, maybe also setup files. Universal application.

4.8 Power Awareness

I will find them in sort of root files of the application, maybe also setup files. Universal application. In the individual features as well which are energy intensive.

4.9 Reduce Size

everywhere service files , API calls, we communicate between server and client. For example, if the data is zipped or not.

4.10 WiFi over Cellular

I will find them in sort of setup, infra files of the application where I define the resources for the application.

4.11 Suppress Logs

I would search for console.log in Javascript in source code, especially the files that are presented to the user and rendered. Also, API call response.

4.12 Batch Operations

Find them in the service file where I know I perform this kind of operation. I will check all API requests. If I have to implement, then I will implement a separate file that all my request go through it

4.13 Cache

Check the files where I know the website is rendered. Maybe I will find something is just cached, instead of reloading. I can find if something is rendered.

4.14 Decrease Rates

Service files where I make backend requests or where I call these services and look they are not repetitive. If I have to implement it, I will implement it in the service file.

4.15 User Knows Best

look at the user features or user setting files and you will see if they can customize it or not.

4.16 Inform Users

have to think about energy intensive operation and find where these actions are implemented. I will check UI files, website file, global files (for alert which can be displayed to user).

4.17 Enough Resolution

component where the website is rendered .

4.17 Sensor Fusion

I have to look at all the instances of sensor usage. I feel it is hard to detect.

4.19 Kill abnormal Tasks

wherever we implement API calls, check if there is a universal setting for time outs, configure , or if there is a timeout after a certain time.

4.20 No Screen Interaction

I do not know. I have never done it.

4.21 Avoid Extraneous Graphics and Animations

graphics display part, where the page is rendered. Think about the cases where the image size can be reduced.

4.22 Manual Sync, On Demand

everywhere where the user synchronizes, makes requests to the server, where the requests are called.

P6: Question: Where in the source code can you find these energy patterns? Please describe your approach to searching each pattern.

I will analyze the source code. I will look at the names of the package, classes, and methods and based on their naming, will look for the patterns, where you would expect them. For example, if there are some network-related names, you would search for reduced size pattern or anti-pattern. I will do an energy review. For some patterns, it can be obvious where to look.
I will also look at how design patterns are detected., and how energy patterns are detected in Android application, like EcoAndroid detects.
the linters which are available in IDEs should include features to look at energy-related improvements or patterns and suggest to users to improve.

4.1 Dark UI Colors

In MVC, will look at the view component of MVC

4.2 Dynamic Retry Delay

Will look at where the communication with Client REST APIs, or if there is a public subscribe method.

4.3 Avoid Extraneous Work

when the data is loaded and presented to the client.

4.4 Race-to-idle

API calls on the client side

4.5 Open Only When Necessary

API calls on the client side, sensor (camera) interactions, database, and graphic elements are shown

4.6 Push over Poll

communication between client and server, how the requests are initiated and handled

4.7 Power Save Mode

Representation (Animations, navigation, graphics)

4.8 Power Awareness

I am not sure how we can read this information on the client side. Power awareness according to my opinion in the controller and view part for MVC. If the graphic is complex, we can see which are energy greedy operation
The pattern should be restricted to the architecture of the application as it can vary a lot.

4.9 Reduce Size

Network traffic, REST APIs (head and body)

4.10 WiFi over Cellular

Only if the website is opened on the mobile, In mobile application, I would look at the communication between front end and backend happens.

4.11 Suppress Logs

I would search for logging libraries and logging commands.

4.12 Batch Operations

database-related operations, user requests in view, and controller for MVC. If the page is loaded and one sees a lot of requests. I would search large files in the source code, there is the possibility that certain things are done in these files and can be optimized for batch operations.

4.13 Cache

Client-side code. I would look at the usage of APIs to see when they have been called, and if the same API has been called multiple times with the same parameters then you can see the opportunity to use the cache mechanism. But also look at the cache mechanism to see whether it has been used in the right context or not.

4.14 Decrease Rates

Nework related components, display related components

4.15 User Knows Best & 4.16 Inform Users

If there are tools that recognize energy patterns then we can know which code is energy greedy or under which features these energy greedy parts are. It depends on the website and its aim. If it is a webpage with movie content, you can look at movie

4.17 Enough Resolution

4.17 Sensor Fusion

4.19 Kill abnormal Tasks

4.20 No Screen Interaction

4.21 Avoid Extraneous Graphics and Animations

4.22 Manual Sync, On Demand