FLOOD MONITORING

PHASE-4

NAME: A. Weis meurial gifta

ROLL NO:210521106059

DOMAIN: IOT

DEFINITION:

Early flood detector, it can enhance the functionality and user experience by incorporating web development technologies. Here's how can integrate web technologies into various aspects of the project: .

- ***Html/css:Design the dashboard's layout and style using HTML and CSS.***
- ***Java script: Implement interactivity for real-time updates, charts, and user management.***
- ***Web framework:can use popular frameworks like React, Angular, or Vue.js for a organized and responsive interface.***

UPDATES AND REPORTING:

❖ *Develop Backend APIs:*

Create a set of API endpoints on your server to handle various functionalities of the early smart flood monitoring such as user authentication, parking spot availability, reservations, and

payments. You can use a web framework like Express.js (Node.js) or Django (Python) to develop these APIs.

❖ *User data analysis:*

Use web development technologies to ensure real-time updates on parking spot availability, reservation confirmation, and payment status. You can achieve this with technologies like WebSocket for real-time communication between the server and clients.
WebSocket: Implement WebSocket communication to push real-time updates to the web and mobile clients when a parking spot's status changes.

❖ *Parking Spot Availability:*

Develop an API endpoint to provide real-time information about flood updatespot availability.

❖ *Reservations:*

Create APIs for reserving flood monitoring. When a user selects a spot and reserves it, the mobile app should send a request to the reservation API.
Implement logic to check spot availability and confirm the reservation.
Return a response .

❖ *API Integration:*

Use HTTP requests (e.g., GET, POST, PUT, DELETE) in the mobile app to communicate with the backend APIs.
Handle API responses in the app to update the user interface and provide feedback to the user.

❖ *Testing and Debugging:*

Test the authenticating functionality by creating test scenarios and debugging any issues that arise.
Verify that the app can interact seamlessly with the backend APIs.

❖ *Deployment:*

Deploy the mobile app to app stores (Google Play Store and Apple App Store) for public use.

❖ *User Support and Updates:*

Provide ongoing support and maintenance for the mit app
Implement updates as needed, addressing user feedback and making improvements.

PROGRAM*:*

Creating a complete mobile app for an IoT early flood monitoringSystem is a complex task that requires a significant amount of code and development effort. I can provide you with a simplified example of a Python program using the Kivy framework to create a basic user interface for a mobile app. Please note that this example is a basic starting point, and it would need to extend it significantly to implement the full functionality of the Smart flood watering System.

CODE:

```
<!DOCTYPE HTML>

<

   Hielo by TEMPLATED

   templated.co @templatedco

   Released for free under the Creative Commons Attribution 3.0
license (templated.co/license)

-->

<html>

   <head>

      <title>DISASTER MANAGEMENT</title>

      <meta charset="utf-8" />
```

```
        <meta name="viewport" content="width=device-width, initial-
scale=1" />
        <link rel="stylesheet" href="assets/css/main.css" />
    </head>
    <body class="subpage">

        <!-- Header -->
            <header id="header">
                <div class="logo"><a href="index.html">DISASTER
MANAGEMENT</a></div>
                <a href="#menu">Menu</a>
            </header>

        <!-- Nav -->
            <nav id="menu">
                <ul class="links">
                    <li><a href="index.html">Home</a></li>
                    <li><a href="https://www.accuweather.com/">Weather
Report</a></li>
                    <li><a
href="https://www.theweathernetwork.com/maps/current-
weather">current news</a></li>
                </ul>
            </nav>


        <!-- One -->
            <section id="One" class="wrapper style3">
                <div class="inner">
                    <header class="align-center">
                        <h2>FLOOD PREPARATION</h2>
                    </header>
                </div>
            </section>


        <!-- Two -->
            <section id="two" class="wrapper style2">
                <div class="inner">
                    <div class="box">
                        <div class="content">
                            <!--<header class="align-center">
                                <p>maecenas sapien feugiat ex purus</p>
                                <h2>Lorem ipsum dolor</h2>
                            </header>-->
                            <img src="images/flood.jpg" width="100%"
height="500px"></br></br>
```

Floods

</br>

- ***Failing to evacuate flooded areas, entering flood waters, or remaining after a flood has passed can result in injury or death. Flooding is a temporary overflow of water onto land that is normally dry. Floods are the most common natural disaster in the United States. Floods may:***

<ul>

<li>Result from rain, snow, coastal storms, storm surges, and overflows of dams and other water systems.</li>

<li>Develop slowly or quickly – Flash floods can come with no warning.</li>

<li>Cause outages, disrupt transportation, damage buildings, and create landslides.</li>

</ul>

- ***This code provides a very basic user interface for the flood monitoring System. For a complete app, that would need to design more advanced UI components, implement user authentication, handle responses from the server, and manage the app's navigation flow.***
- ***Additionally, for a production-ready app, that might want to consider using a dedicated cross-platform mobile app development framework like React Native, Flutter, or others, as they offer a more robust and scalable approach to mit app development.***