Smart Parking System Coding Standards

Overview

Our Smart Parking System aims to provide efficient parking management using Flutter for the frontend, Node.js and Python for the backend, and a variety of modern technologies and frameworks. This document outlines the coding standards and guidelines for ensuring consistent, readable, and maintainable code across the project.

Frontend Development

Framework

We use Flutter for developing the frontend. Flutter allows us to create natively compiled applications for mobile, web, and desktop from a single codebase.

Components Library

Use Flutter Widgets for creating the user interface. Ensure consistent styling and theming throughout the application.

File and Folder Naming Conventions

File and folder names should be in snake_case.

Correct

```
plaintext
Copy code
my_widget.dart
my_folder/
```

Incorrect

```
plaintext
Copy code
myWidget.dart
MyFolder/
```

Dart

1. Use UpperCamelCase for class names.

Correct

```
dart
Copy code
class MyClass {
  // Class implementation
}
```

Incorrect

```
dart
Copy code
class myClass {
  // Class implementation
}
```

2. Use lowerCamelCase for variables and functions.

Correct

```
dart
Copy code
int myVariable = 10;

void myFunction() {
  // Function implementation
}
```

Incorrect

```
dart
Copy code
int MyVariable = 10;

void MyFunction() {
  // Function implementation
}
```

3. Prefer using const and final over var when possible.

Correct

```
dart
Copy code
const int myConstant = 5;
final String myFinalVariable = 'Hello';
```

Incorrect

```
dart
Copy code
var myVariable = 5;
```

4. Avoid using magic strings and numbers; use constants instead.

Correct

```
dart
Copy code
const String apiUrl = 'https://api.example.com';
```

Incorrect

```
dart
Copy code
String url = 'https://api.example.com';
```

Backend Development

Node.js

API Framework

We use Express with NestJS for building our APIs.

File and Folder Naming Conventions

File and folder names should be in kebab-case.

Correct

```
plaintext
Copy code
my-controller.js
my-service/
```

Incorrect

```
plaintext
Copy code
myController.js
MyService/
```

JavaScript

1. Use UpperCamelCase for class names and lowerCamelCase for variables and functions.

Correct

```
javascript
Copy code
class MyController {
  constructor() {
    this.myService = new MyService();
  }

  myFunction() {
    // Function implementation
  }
}
```

Incorrect

```
javascript
Copy code
class myController {
  constructor() {
    this.MyService = new MyService();
  }

  MyFunction() {
    // Function implementation
  }
}
```

2. Use const and let instead of var.

Correct

```
javascript
Copy code
const myConstant = 5;
let myVariable = 10;
```

Incorrect

```
javascript
Copy code
var myVariable = 10;
```

3. Use arrow functions for anonymous functions.

Correct

```
javascript
Copy code
const myFunction = () => {
  // Function implementation
}
```

Incorrect

```
javascript
Copy code
const myFunction = function() {
  // Function implementation
}
```

4. Throw errors instead of strings or objects.

Correct

```
javascript
Copy code
if (param <= 0) {
  throw new Error('Parameter must be greater than 0');
}
```

Incorrect

```
javascript
Copy code
if (param <= 0) {
  throw 'Parameter must be greater than 0';
}
```

Python

API Framework

We use Django for building Python-based APIs.

File and Folder Naming Conventions

File and folder names should be in snake_case.

Correct

```
plaintext
Copy code
my_module.py
my_package/
```

Incorrect

```
plaintext
Copy code
myModule.py
MyPackage/
```

Python Coding Standards

1. Use UpperCamelCase for class names and lower_snake_case for variables and functions.

Correct

```
python
Copy code
class MyClass:
    def __init__(self):
        self.my_variable = 10

    def my_function(self):
        # Function implementation
```

Incorrect

```
python
Copy code
class myClass:
    def __init__(self):
        self.myVariable = 10

    def MyFunction(self):
        # Function implementation
```

2. Use const for constants.

Correct

```
python
Copy code
MY_CONSTANT = 5
```

Incorrect

```
python
Copy code
myConstant = 5
```

3. Follow PEP 8 guidelines for code formatting.

Correct

```
python
Copy code
def my_function(param1, param2):
    if param1 > param2:
        return param1
    else:
        return param2
```

Incorrect

```
python
Copy code
def my_function(param1,param2):
    if(param1>param2): return param1
    else:return param2
```

Machine Learning

Frameworks

We use TensorFlow and PyTorch for machine learning models, specifically for parking availability prediction.

Python Coding Standards for Machine Learning

1. Use clear and descriptive variable names.

Correct

```
python
Copy code
model = tf.keras.Sequential()
```

Incorrect

```
python
Copy code
m = tf.keras.Sequential()
```

2. Document your code and include comments where necessary.

Correct

```
python
Copy code
# Define the model
model = tf.keras.Sequential()
```

Incorrect

```
python
Copy code
model = tf.keras.Sequential()
```

Data Store

Database

We use MongoDB for data storage.

Naming Conventions

1. Use camelCase for field names.

Correct

```
json
Copy code
{
  "userName": "JohnDoe",
  "userEmail": "john@example.com"
}
```

Incorrect

```
json
Copy code
{
  "user_name": "JohnDoe",
  "user_email": "john@example.com"
}
```

Version Control

Git

We use GitHub for version control and team organization.

Branching Model

Follow the conventional Gitflow Workflow.

Commit Messages

Ensure commit messages are descriptive.

Correct

```
arduino
Copy code
"Refactor the conditional logic in handleRemoveImage function for clarity"
```

Incorrect

```
arduino
Copy code
"Updated function"
```

Containerization

Docker

We use Docker for containerization.

Naming Conventions

Use `snake_case` for Docker container names and image tags.

Correct

```
plaintext
Copy code
my_service_image
```

Incorrect

```
plaintext
Copy code
myServiceImage
```

Linting

Linting Tool

We use flutter analyze for linting our Flutter code.

Configuration

- Ensure flutter analyze is run on every commit to maintain code quality.
- Configure flutter analyze to enforce the coding standards and guidelines specified in this document.

Conclusion

By adhering to these coding standards, we aim to ensure consistency, readability, and maintainability across our Smart Parking System project. Consistent coding practices also facilitate easier collaboration among team members and contribute to the overall quality of our application.