10 ChatGPT prompts for Software Engineers

By vmixvietnam.

1. Generate boilerplate code

Prompt formula:

Generate boilerplate code for an app that [***explain what you need this app to do***]. Please use [***explain what languages and frameworks should be used***].

Prompt example:

Generate boilerplate code for an app that integrates to an external API. Please use javascript code on the express.js framework.

2. Compare frameworks/algorithms

Prompt formula:

I'm building a new [***explain what you're building***], and want to compare [***first comparison item***] with [***second comparison item***]. Please propose the scope for a simple [***what you're building***], and generate two code bases that fulfill that scope, one using[***first comparison item***] and another using [***second comparison item***]. Please redact clear instructions for me to run both apps on my local machine.

Prompt example:

I'm building a new frontend app, and want to compare React.js with Vue.js. Please propose the scope for a simple frontend app, and generate two code bases that fulfill that scope, one using React.js and another using Vue.js. Please redact clear instructions for me to run both apps on my local machine.

3. Explain code

Prompt formula:

Explain this code to me:

[***code you want to be explained***]

Prompt example:

Explain this code to me:

```
fetch("https://api.stripe.com/v1/payments?created[gte]=last_month")
.then(response => response.json())
.then(data => {
let transactions = data.transactions;
let groupedTransactions = {};
transactions.forEach(transaction => {
if (!groupedTransactions[transaction.entity]) {
groupedTransactions[transaction.entity] = transaction.amount;
} else {
groupedTransactions[transaction.entity] += transaction.amount;
}
});
let sortedTransactions = Object.entries(groupedTransactions).sort((a, b) => b[1] - a[1]);
sortedTransactions.forEach(transaction => {
console.log(${transaction[0]}: $${Math.ceil(transaction[1])});
});
});
}
```

4. Comment code

Prompt formula:

Regenerate the code snippet below, but please include comments to each line of code:

[***code to be commented***]

Prompt example:

Regenerate the code snippet below, but please include comments to each line of code:

```
fetch("https://api.stripe.com/v1/payments?created[gte]=last_month")
.then(response => response.json())
.then(data => {
let transactions = data.transactions;
let groupedTransactions = {};
transactions.forEach(transaction => {
if (!groupedTransactions[transaction.entity]) {
groupedTransactions[transaction.entity] = transaction.amount;
} else {
groupedTransactions[transaction.entity] += transaction.amount;
}
});
let sortedTransactions = Object.entries(groupedTransactions).sort((a, b) => b[1] - a[1]);
sortedTransactions.forEach(transaction => {
console.log(${transaction[0]}: $${Math.ceil(transaction[1])});
});
});
}
```

5. Generate test cases

Prompt formula:

Write test cases for [***cases to be tested***] to the below code snippet. First outline the test cases you'll write. Second, write the test cases in [***language and framework to be used to write the tests***].

[***code to be tested***]

Prompt example:

Write test cases for the main edge cases that could happen to the below code snippet. First outline the test cases you'll write. Second, write the test cases in javascript using the Jest framework.

```
fetch("https://api.stripe.com/v1/payments?created[gte]=last_month")
.then(response => response.json())
.then(data => {
let transactions = data.transactions;
let groupedTransactions = {};
transactions.forEach(transaction => {
if (!groupedTransactions[transaction.entity]) {
groupedTransactions[transaction.entity] = transaction.amount;
} else {
groupedTransactions[transaction.entity] += transaction.amount;
}
});
let sortedTransactions = Object.entries(groupedTransactions).sort((a, b) => b[1] - a[1]);
sortedTransactions.forEach(transaction => {
console.log(${transaction[0]}: $${Math.ceil(transaction[1])});
});
});
}
```

6. Generate documentation

Prompt formula:

Generate documentation for the code below. You should include detailed instructions to allow a developer to run it on a local machine, explain what the code does, and list vulnerabilities that exist in this code.

[***code to be documented***]

Prompt example:

Generate documentation for the code below. You should include detailed instructions to allow a developer to run it on a local machine, explain what the code does, and list vulnerabilities that exist in this code.

```
fetch("https://api.stripe.com/v1/payments?created[gte]=last_month")
.then(response => response.json())
.then(data => {
let transactions = data.transactions;
let groupedTransactions = {};
transactions.forEach(transaction => {
if (!groupedTransactions[transaction.entity]) {
groupedTransactions[transaction.entity] = transaction.amount;
} else {
groupedTransactions[transaction.entity] += transaction.amount;
}
});
let sortedTransactions = Object.entries(groupedTransactions).sort((a, b) => b[1] - a[1]);
sortedTransactions.forEach(transaction => {
console.log(${transaction[0]}: $${Math.ceil(transaction[1])});
});
});
}
```

7. Generate regexes

Prompt formula:

Generate a regex to match [***the pattern you want to match***]

Prompt example:

Generate a regex to match an email address

8. Rewrite code using correct style

Prompt formula:

Rewrite the code below following the [***guidelines to be followed***].

[***code you want to rewrite***]

Prompt example:

Rewrite the code below following the Google style guidelines for javascript.

```
fetch("https://api.stripe.com/v1/payments?created[gte]=last_month")
.then(response => response.json())
.then(data => {
let transactions = data.transactions;
let groupedTransactions = {};
transactions.forEach(transaction => {
if (!groupedTransactions[transaction.entity]) {
groupedTransactions[transaction.entity] = transaction.amount;
} else {
groupedTransactions[transaction.entity] += transaction.amount;
}
});
let sortedTransactions = Object.entries(groupedTransactions).sort((a, b) => b[1] - a[1]);
sortedTransactions.forEach(transaction => {
console.log(${transaction[0]}: $${Math.ceil(transaction[1])});
});
});
}
```

9. Find bugs in code

Prompt formula:

Please find the bug in the code below. This is what it should be doing:
[***outline of the desired functionality***]

Code:
[***code to be debugged***]

Prompt example:

Please find the bug in the code below. This is what it should be doing:
1. Fetch the response from the stripe API for payments received last month.
2. Parse the response json into an arrays with all transactions.
3. Traverse the array to group all transactions from the same entity, and sums their amounts. The result is stored in a different array.
3. Sort the resulting array by amount received, descending.
4. Write to the console all payments, sorted by date, with money amounts rounded up to the integer.

Code:

```
fetch("https://api.stripe.com/v1/payments?created[gte]=last_month")
.then(response => response.json())
.then(data => {
let transactions = data.transactions;
let groupedTransactions = {};
transactions.forEach(transaction => {
if (groupedTransactions[transaction.entity]) {
groupedTransactions[transaction.entity] = transaction.amount;
} else {
groupedTransactions[transaction.entity] += transaction.amount;
}
});
let sortedTransactions = Object.entries(groupedTransactions).sort((a, b) => b[1] - a[1]);
sortedTransactions.forEach(transaction => {
console.log(${transaction[0]}: $${Math.ceil(transaction[1])});
});
});
}
```

10. Solve leetcode type algorithms

Prompt formula:

Generate code in [***desired language***] to solve the following challenge:
[***outline of the challenge to be solved***]

Prompt example:

Generate code in javascript to solve the following challenge:

- We have one 2D array, filled with zeros and ones.
- We have to find the starting point and ending point of all rectangles filled with 0.
- It is given that rectangles are separated and do not touch each other however they can touch the boundary of the array.
- A rectangle might contain only one element.

Example array:

Input = [[1, 1, 1, 1, 1, 1, 1], [1, 1, 1, 1, 1, 1, 1], [1, 1, 1, 0, 0, 0, 1], [1, 1, 1, 0, 0, 0, 1], [1, 1, 1, 1, 1, 1, 1], [1, 1, 1, 1, 1, 1, 1], [1, 1, 1, 1, 1, 1, 1], [1, 1, 1, 1, 1, 1, 1]]

Final note

Please use your best judgment and critical thinking when reviewing other's code (either your colleagues, someone on the internet or generative AI tools). ChatGPT can provide incomplete or wrong answers. Do NOT push any responses to prod before thoroughly reviewing them.