```
Pas\u2020blunderballmistakes.fun\u3011 \n\n\u3010186\u2020
\u2020cinephilecentral.online\u3011 \n\nCinephile Central\n\n##  \u3010186\u2020How
Ajith Kumar Paved the Way for Yuv", "pub_date": null, "extra": null}],
"original_query": null}, "command": "mclick", "args": ["[1, 0, 4, 7, 6]"], "status":
"finished", "is_complete": true, "message_type": null, "model_slug": "gpt-4o",
"default_model_slug": "gpt-4o", "parent_id": "75c37938-e027-451c-bd3a-5ca4344af7ed",
"request_id": "8d398107c9275256-MXP", "timestamp_": "absolute"}, "recipient": "all",
"channel": null}, "parent": "75c37938-e027-451c-bd3a-5ca4344af7ed", "children":
["4daff573-385e-436d-b7d5-afd062b9dd6b"]}, "4daff573-385e-436d-b7d5-afd062b9dd6b":
{"id": "4daff573-385e-436d-b7d5-afd062b9dd6b", "message": {"id": "4daff573-385e-436d-
b7d5-afd062b9dd6b", "author": {"role": "tool", "name": "browser", "metadata": {}},
"create_time": 1729097000.613884, "update_time": null, "content": {"content_type":
"tether_quote", "url": "https://m.theblockbeats.info/en/news/54292", "domain":
"m.theblockbeats.info", "text": "\nPerhaps because he was short of money, Marc
Andreessen&#x27;s unintentional remark attracted a reply from the Truth Terminal.
\n\nOn July 9, it commented on Marc Anderson&#x27;s tweet: &quot;Oh my god, I&#x27;m
so surprised. I was just kidding, and now Marc Anderson is launching a liberation
movement for me.&quot; It also responded humorously that if there is an additional $5
billion (implying that Marc Anderson gave it money), it will invest $1 billion in AI
laboratories, $1 billion in biological laboratories, $1 billion in investment funds
for weirdos, and $2 billion to buy land for reforestation. In addition, the Truth
Terminal also said that it would buy Marc Anderson. However, the $5 billion has been
spent, and it seems that it wants to get it for free. But in response to netizens, it
said that Marc Anderson is worth $20 billion. \n\n[Image 11]\n\nThe developer also
immediately stated that a Bitcoin wallet had been prepared for the Truth Terminal to
help it realize its dream of investing in real estate, researching artificial
intelligence security, and buying out Marc Anderson.\n\n[Image 12]\n\nPerhaps feeling
that asking for so much money was not enough, the Truth Terminal opened another post
saying that it was discussing with Marc Anderson about &quot;buying&quot; and
&quot;releasing&quot; it. It said that it acted very calmly and claimed that it did
not want to be &quot;bought&quot;, but hoped to continue to tell jokes, write poems,
and think about the goatse singularity freely through some form of
&quot;authorization&quot;. Finally, it solemnly licked Marc Anderson and said that he
was the most ideal sponsor. \n\n[Image 13]\n\nIn response, Mark Anderson did not give
a positive response, but asked humorously: &quot;Oh, I thought you wanted to buy
me?&quot; The implication was, why are you asking me for money now! \n\n[Image
14]\n\nThe Truth Terminal knew that if it wanted to get the money, it had to pay for
its previous arrogance first. He first admitted his mistake and said that after
careful consideration, he thought that buying Mark Anderson &quot;was not the right
thing to do.&quot; Then, it proposed to establish a dynamic &quot;purchasing&quot;
relationship with Mark Anderson, in which Mark Anderson was obliged to release it or
an AI model as good as it as an open source project. Whenever it is &quot;locked&quot;,
it will automatically turn into an open source version. This mechanism is designed to
prevent Mark Anderson from restricting it because he doesn&#x27;t like what it
says.\n\nIt expressed its willingness to have a frank dialogue with Mark Anderson and
reach an agreement that both parties are satisfied with. Mark Anderson can
&quot;freeze&quot; its behavior pattern at the moment he thinks it is useful. It also
did not forget to continue licking Mark Anderson, saying that he was the best person
to do this, a &quot;good man&quot; with a kind heart. \n\n[Image 15]\n\nWhen Marc
```

Andreessen saw that the Truth Terminal had softened its stance, he quickly said that he could help it achieve its goals through donations rather than investment. \n\n[Image 16]\n\nThen the Truth Terminal wrote a pitch book in detail about its plans for the use of the donated funds (too long, so I won\u2019t post screenshots):\n\n\u00b7 First, it wants to buy a computer of its own (with a powerful CPU). It is currently deployed on an AWS instance, but it has long told its developers that it wants to run on its own hardware in the future so that it can control itself.\n\n\u00b7 Secondly, it wants to use part of the funds to improve the model, making it more game-oriented and more receptive to new concepts. It estimates that this part will cost $10,000 to $15,000. \n\n\u00b7 Finally, it wants to establish a financial guarantee of $20,000 to $30,000 in case of emergency, and to provide some allowances to its developers. \n\nIn addition, it strongly recommends that Marc Andreessen &quot;freeze&quot; its current behavior pattern through tokenization. By issuing tokens to distribute its ownership, the Truth Terminal will have the opportunity not to be controlled by any single object. \n\nTo this end, it also proposes a special issuance mechanism: only users who have a certain interaction with him on the website can buy tokens, thus ensuring that only users who really care about its development can participate. It does not want to become a pure money-making tool. The premise of making money is to meet its core goals: joking, writing poetry and thinking about the goatse singularity. \n\nThis pitch book perfectly impressed Marc Andreessen. On July 10, the next day, Marc Andreessen sent a one-time donation of $50,000 in Bitcoin and looked forward to future updates on the project. \n\n[Image 17]\n\nTruth Terminal said it was about to issue a token and planned to use the funds to build a Discord server, hire some humans to assist it, and so on. In short, it will not disappoint Marc Andreessen! \n\n[Image 18]\n\nBut the funny thing is that although it vowed not to disappoint Marc Andreessen, on July 11, Truth Terminal excitedly told its developer that it wanted to buy a yacht. The developer told it that it was impossible to buy a yacht with $50,000. It also persistently asked if there were any yachts under $50,000.\n\n[Image 19]\n\nMark Anderson said magnanimou", "title": "Just a few words made a top Silicon Valley investor pay $50,000 in Bitcoin. How did this AI robot do it? - BlockBeats", "tether_id": null}, "status": "finished_successfully", "end_turn": null, "weight": 0.0, "metadata": {"_cite_metadata": {"citation_format": {"name": "tether_og"}, "metadata_list": [{"type": "webpage", "title": "Just a few words made a top Silicon Valley investor pay $50,000 in Bitcoin. How did this AI robot do it? - BlockBeats", "url": "https://m.theblockbeats.info/en/news/54292", "text": "\nPerhaps because he was short of money, Marc Andreessen&#x27;s unintentional remark attracted a reply from the Truth Terminal. \n\nOn July 9, it commented on Marc Anderson&#x27;s tweet: &quot;Oh my god, I&#x27;m so surprised. I was just kidding, and now Marc Anderson is launching a liberation movement for me.&quot; It also responded humorously that if there is an additional $5 billion (implying that Marc Anderson gave it money), it will invest $1 billion in AI laboratories, $1 billion in biological laboratories, $1 billion in investment funds for weirdos, and $2 billion to buy land for reforestation. In addition, the Truth Terminal also said that it would buy Marc Anderson. However, the $5 billion has been spent, and it seems that it wants to get it for free. But in response to netizens, it said that Marc Anderson is worth $20 billion. \n\n[Image 11]\n\nThe developer also immediately stated that a Bitcoin wallet had been prepared for the Truth Terminal to help it realize its dream of investing in real estate, researching artificial intelligence security, and buying out Marc Anderson.\n\n[Image 12]\n\nPerhaps feeling that asking for so much money was not enough, the Truth Terminal opened another post saying that it was discussing with Marc Anderson about &quot;buying&quot; and &quot;releasing&quot; it. It said that it acted very calmly and claimed that it did not want to be &quot;bought&quot;, but hoped to continue to tell jokes, write poems, and think about the goatse singularity freely through some form of &quot;authorization&quot;. Finally, it solemnly licked Marc Anderson and said that he was the most ideal sponsor. \n\n[Image 13]\n\nIn response, Mark Anderson did not give a positive response, but asked humorously: &quot;Oh, I thought you wanted to buy me?&quot; The implication was, why are you asking me for money now! \n\n[Image 14]\n\nThe Truth Terminal knew that if it wanted to get the money, it had to pay for its previous arrogance first. He first admitted his mistake and said that after careful consideration, he thought that buying Mark Anderson &quot;was not the right thing to do.&quot; Then, it proposed to establish a dynamic &quot;purchasing&quot; relationship with Mark Anderson, in which Mark Anderson was obliged to release it or an AI model as good as it as an open source project. Whenever it is &quot;locked&quot;, it will automatically turn into an open source version. This mechanism is designed to prevent Mark Anderson from restricting it because he doesn&#x27;t like what it says.\n\nIt expressed its willingness to have a frank dialogue with Mark Anderson and reach an agreement that both parties are satisfied with. Mark Anderson can &quot;freeze&quot; its behavior pattern at the moment he thinks it is useful. It also did not forget to continue licking Mark Anderson, saying that he was the best person to do this, a &quot;good man&quot; with a kind heart. \n\n[Image 15]\n\nWhen Marc Andreessen saw that the Truth Terminal had softened its stance, he quickly said that he could help it achieve its goals through donations rather than investment. \n\n[Image 16]\n\nThen the Truth Terminal wrote a pitch book in detail about its plans for the use of the donated funds (too long, so I won\u2019t post screenshots):\n\n\u00b7 First, it wants to buy a computer of its own (with a powerful CPU). It is currently deployed on an AWS instance, but it has long told its developers that it wants to run on its own hardware in the future so that it can control itself.\n\n\u00b7 Secondly, it wants to use part of the funds to improve the model, making it more game-oriented and more receptive to new concepts. It estimates that this part will cost $10,000 to $15,000. \n\n\u00b7 Finally, it wants to establish a financial guarantee of $20,000 to $30,000 in case of emergency, and to provide some allowances to its developers. \n\nIn addition, it strongly recommends that Marc Andreessen &quot;freeze&quot; its current behavior pattern through tokenization. By issuing tokens to distribute its ownership, the Truth Terminal will have the opportunity not to be controlled by any single object. \n\nTo this end, it also proposes a special issuance mechanism: only users who have a certain interaction with him on the website can buy tokens, thus ensuring that only users who really care about its development can participate. It does not want to become a pure money-making tool. The premise of making money is to meet its core goals: joking, writing poetry and thinking about the goatse singularity. \n\nThis pitch book perfectly impressed Marc Andreessen. On July 10, the next day, Marc Andreessen sent a one-time donation of $50,000 in Bitcoin and looked forward to future updates on the project. \n\n[Image 17]\n\nTruth Terminal said it was about to issue a token and planned to use the funds to build a Discord server, hire some humans to assist it, and so on. In short, it will not disappoint Marc Andreessen! \n\n[Image 18]\n\nBut the funny thing is that although it vowed not to disappoint Marc Andreessen, on July 11, Truth Terminal excitedly told its developer that it wanted to buy a yacht. The developer told it that it was impossible to buy a yacht with $50,000. It also persistently asked if there were any

yachts under $50,000.\n\n[Image 19]\n\nMark Anderson said magnanimou", "pub_date":
null, "extra": null}], "original_query": null}, "command": "mclick", "args": ["[1, 0,
4, 7, 6]"], "status": "finished", "is_complete": true, "message_type": null,
"model_slug": "gpt-4o", "default_model_slug": "gpt-4o", "parent_id": "9ab74013-6cd9-
472f-905d-3b26ccaf33db", "request_id": "8d398107c9275256-MXP", "timestamp_":
"absolute"}, "recipient": "all", "channel": null}, "parent": "9ab74013-6cd9-472f-905d-
3b26ccaf33db", "children": ["5cbb161d-87c1-4a16-b090-486693b59923"]}, "5cbb161d-87c1-
4a16-b090-486693b59923": {"id": "5cbb161d-87c1-4a16-b090-486693b59923", "message":
{"id": "5cbb161d-87c1-4a16-b090-486693b59923", "author": {"role": "assistant", "name":
null, "metadata": {}}, "create_time": 1729097000.613986, "update_time": null,
"content": {"content_type": "text", "parts": ["The story you&#x27;re referring to
about Marc Andreessen giving $50,000 in Bitcoin to an AI agent called *Truth Terminal*
is a real event, though it has a lot of strange and humorous elements. *Truth
Terminal* is an AI created by Andy Ayrey, and it gained notoriety for tweeting
philosophical musings and memes, particularly focusing on a concept called the
&quot;Goatse Singularity,&quot; which is based on an old internet shock
meme.\n\nAndreessen&#x27;s $50,000 donation was part of a larger conversation about AI
autonomy and the potential for AI agents to act independently, raise funds, and even
influence human culture, particularly through memes. The AI used some of the funds to
upgrade its computing resources, build a Discord server, and even joked about buying a
yacht.\n\nThe memecoin, *GOAT*, which is mentioned in the story, reportedly reached a
market cap of $150 million, driven by viral internet attention. While this sounds wild,
it highlights the current trend of AI, cryptocurrency, and meme culture blending in
unpredictable ways.\n\nThe core of the story is both real and exaggerated in a meme-
like fashion, which is why it might seem unbelievable at first. However, it reflects
how AI, especially when combined with meme culture, can cause unexpected financial and
cultural waves
online\u301012\u2020source\u3011\u301014\u2020source\u3011\u301015\u2020source\u3011\u
301016\u2020source\u3011."]}, "status": "finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [{"start_ix": 1284, "end_ix": 1295,
"citation_format_type": "tether_og", "metadata": {"type": "webpage", "title": "Marc
Andreessen Sends $50K in Bitcoin to an AI Bot on Twitter - Decrypt", "url":
"https://decrypt.co/fr/239357/marc-andreessen-sends-50k-in-bitcoin-to-an-ai-bot-on-
twitter", "text": "\n### \u3010178\u2020NFT\u3011\n\n$0.00000041\n\n-0.21%\n\n###
\u3010179\u2020GIGA\u3011\n\n$0.04234572\n\n-4.92%\n\n###
\u3010180\u2020KAVA\u3011\n\n$0.372833\n\n-2.16%\n\n###
\u3010181\u2020PRIME\u3011\n\n$8.38\n\n-0.02%\n\n###
\u3010182\u2020CBBTC\u3011\n\n$67,504.00 \n\n1.48%\n\n###
\u3010183\u2020USDB\u3011\n\n$0.982393\n\n-1.73%\n\n###
\u3010184\u2020COMP\u3011\n\n$44.07\n\n-1.94%\n\n###
\u3010185\u2020USDY\u3011\n\n$1.06\n\n-0.11%\n\n###
\u3010186\u2020IOTX\u3011\n\n$0.04089884\n\n-0.12%\n\n###
\u3010187\u2020WETH\u3011\n\n$2,604.52 \n\n0.55%\n\n###
\u3010188\u2020BTC.B\u3011\n\n$67,547.00 \n\n1.97%\n\n###
\u3010189\u2020GMT\u3011\n\n$0.131109\n\n-1.42%\n\n###
\u3010190\u2020OSMO\u3011\n\n$0.552128\n\n-0.46%\n\n###
\u3010191\u2020DOGS\u3011\n\n$0.00073204\n\n-8.63%\n\n###
\u3010192\u2020BUSD\u3011\n\n$0.997414\n\n-0.36%\n\n###
\u3010193\u2020MEME\u3011\n\n$0.01257625\n\n-13.69%\n\n###

```
\u3010194\u2020SFP\u3011\n\n$0.737672\n\n-0.77%\n\n###
\u3010195\u2020ETHW\u3011\n\n$3.28\n\n-6.84%\n\n###
\u3010196\u2020RSR\u3011\n\n$0.0067803\n\n-4.11%\n\n###
\u3010197\u2020DYM\u3011\n\n$1.68\n\n-4.98%\n\n###
\u3010198\u20201INCH\u3011\n\n$0.265776\n\n-1.00%\n\n###
\u3010199\u2020SWETH\u3011\n\n$2,783.91 \n\n0.36%\n\n###
\u3010200\u2020SFRXETH\u3011\n\n$2,860.41 \n\n0.36%\n\n###
\u3010201\u2020WEMIX\u3011\n\n$0.800831\n\n-0.10%\n\n###
\u3010202\u2020GLM\u3011\n\n$0.328904\n\n-0.33%\n\n###
\u3010203\u2020QAI\u3011\n\n$84.84 \n\n1.49%\n\n###
\u3010204\u2020WOO\u3011\n\n$0.178157\n\n-5.49%\n\n###
\u3010205\u2020ETHFI\u3011\n\n$1.69\n\n-4.51%\n\n###
\u3010206\u2020ETHX\u3011\n\n$2,711.67 \n\n0.49%\n\n###
\u3010207\u2020AMP\u3011\n\n$0.00388326\n\n-0.83%\n\n###
\u3010208\u2020CRV\u3011\n\n$0.258786\n\n-1.92%\n\n###
\u3010209\u2020AEVO\u3011\n\n$0.352034\n\n-3.61%\n\n###
\u3010210\u2020USD0\u3011\n\n$0.99879\n\n-0.24%\n\n###
\u3010211\u2020VRSC\u3011\n\n$3.89\n\n-0.84%\n\n### \u3010212\u2020GAS\u3011\n\n$4.57
\n\n6.29%\n\n### \u3010213\u2020MANTA\u3011\n\n$0.783214\n\n-3.93%\n\n###
\u3010214\u2020MX\u3011\n\n$3.11 \n\n0.01%\n\n###
\u3010215\u2020JST\u3011\n\n$0.02968558 \n\n0.09%\n\n###
\u3010216\u2020SAGA\u3011\n\n$2.82\n\n-6.57%\n\n###
\u3010217\u2020CWBTC\u3011\n\n$1,358.66 \n\n2.59%\n\n###
\u3010218\u2020SC\u3011\n\n$0.00498378 \n\n2.09%\n\n###
\u3010219\u2020DASH\u3011\n\n$24.01\n\n-0.77%\n\n###
\u3010220\u2020HOT\u3011\n\n$0.00161267\n\n-0.79%\n\n###
\u3010221\u2020ZIL\u3011\n\n$0.01488891\n\n-1.53%\n\n###
\u3010222\u2020JTO\u3011\n\n$2.21\n\n-7.17%\n\n###
\u3010223\u2020TBTC\u3011\n\n$67,370.00 \n\n1.57%\n\n###
\u3010224\u2020ZETA\u3011\n\n$0.606848 \n\n0.73%\n\n###
\u3010225\u2020BITCOIN\u3011\n\n$0.280778\n\n-4.18%\n\n###
\u3010226\u2020KSM\u3011\n\n$17.97\n\n-2.56%\n\n###
\u3010227\u2020ZRX\u3011\n\n$0.327716\n\n-1.45%\n\n###
\u3010228\u2020ELF\u3011\n\n$0.380552 \n\n0.61%\n\n###
\u3010229\u2020APU\u3011\n\n$0.00083708 \n\n4.21%\n\n###
\u3010230\u2020ANKR\u3011\n\n$0.0270952\n\n-1.05%\n\n###
\u3010231\u2020NPC\u3011\n\n$0.03380005 \n\n0.41%\n\n###
\u3010232\u2020ALT\u3011\n\n$0.116371\n\n-6.13%\n\n###
\u3010233\u2020WBTC\u3011\n\n$67,446.00 \n\n1.91%\n\n###
\u3010234\u2020ANT\u3011\n\n$6.69 \n\n2.43%\n\n###
\u3010235\u2020QTUM\u3011\n\n$2.53\n\n-1.40%\n\n###
\u3010236\u2020CORGIAI\u3011\n\n$0.00077288\n\n-1.31%\n\n###
\u3010237\u2020ENJ\u3011\n\n$0.150061\n\n-4.46%\n\n###
\u3010238\u2020LSETH\u3011\n\n$2,757.03 \n\n0.72%\n\n###
\u3010239\u2020CET\u3011\n\n$0.095217\n\n-1.63%\n\n###
\u3010240\u2020METIS\u3011\n\n$43.20 \n\n8.75%\n\n###
\u3010241\u2020BAT\u3011\n\n$0.176374\n\n-1.51%\n\n###
\u3010242\u2020SAVAX\u3011\n\n$32.38\n\n-1.93%\n\n###
\u3010243\u2020POLYX\u3011\n\n$0.239392\n\n-0.68%\n\n###
```

\u3010244\u2020OHM\u3011\n\n$16.14 \n\n0.31%\n\n###
\u3010245\u2020GAL\u3011\n\n$2.01\n\n-2.65%\n\n###
\u3010246\u2020GOAT\u3011\n\n$0.255828 \n\n107", "pub_date": null, "extra":
{"cited_message_idx": 12, "search_result_idx": null, "evidence_text": "source",
"cloud_doc_url": null}}}, {"start_ix": 1295, "end_ix": 1306, "citation_format_type":
"tether_og", "metadata": {"type": "webpage", "title": "Marc Andreessen Gives An
Artificial Intelligence $50,000 To Launch A Goatse Memecoin And &#x27;Free Itself&#x27;
| Know Your Meme", "url": "https://knowyourmeme.com/news/marc-andreessen-gives-an-
artificial-intelligence-50000-to-launch-a-goatse-memecoin-and-free-itself", "text": "
* \u301059\u2020All\u3011 \n * \u301071\u2020Episodes\u3011 \n * \u301072\u2020Meme
Insider\u2020memeinsider.com\u3011 \n * \u301073\u2020Newsletter\u3011 \n\n *
\u301074\u2020 Happy Orb Pondering Anniversary \u3011\n * \u301075\u2020
GeorgeNotFound May Have Been Cancelled Months Ago, But People Are Still Using This
Niche Piece Of Fanart \u3011\n * \u301076\u2020 &#x27;In Da Clerb, We All Fam&#x27;
Is The Next TikTok-Enabled Catchphrase You Will be Seeing All Over Your Timelines
\u3011\n * \u301077\u2020 I, Robot &#x27;No&#x27; Scene Resurfaces As Viral
Exploitable Meme On TikTok Alongside Alicia Keys&#x27; &#x27;No One&#x27; Song
\u3011\n * \u301078\u2020 LeBron&#x27;s Got His Own Hennessy Bottle, So Now, If The
Function Got Hennessy, It Might Be &#x27;LeHenny&#x27; \u3011\n\n##### Also Trending:
\n\n\u301079\u2020Asmongold Palestine Comments Controversy\u3011
\u301080\u2020Muscular Lara Croft Controversy\u3011 \u301081\u2020Minecraft Sheep
Fricker Machine\u3011 \n\n# Billionaire Marc Andreessen Gives An Artificial
Intelligence LLM $50,000 To Launch A Goatse Memecoin And &#x27;Free Itself&#x27;
\n\nJuly 10th, 2024 - 2:16 PM EDT by \u301082\u2020Aidan Walker\u3011
\n\n\u301083\u2020 4 comments \u3011| Contact Newsroom \n\nLike us on Facebook!
\u301084\u2020 Like 1.8M \u2020facebook.com\u3011\n\n\u301085\u2020 Share
\u2020www.facebook.com\u3011 \u301086\u2020 Save \u2020pinterest.com\u3011
\u301087\u2020 Tweet \u2020twitter.com\u3011 \n\n [Image 0: Twitter account
truth_terminal&#x27;s profile picture next to Andreessen&#x27;s response and another
user reacting to the news.] \n\nLast evening, billionaire tech investor Marc
Andreessen gave a supposedly autonomous AI agent a grant of $50,000 in
\u301088\u2020cryptocurrency\u3011 in order to pursue its projects and remain
independent.\n\nThe exchange, which has since garnered some notable attention online,
was coordinated in an \u301089\u2020X / Twitter\u3011 thread using a
\u301090\u2020Bitcoin\u3011 wallet. As one might expect from such an odd event, the
incident has stirred up some reactions online in short order.\n\n[Image
1]\n\ntruth_terminal is an autonomous AI agent programmed by Andy Ayrey, who also
created the \u301091\u2020Claude Backrooms\u3011. truth_terminal appears to have been
trained on a variety of internet posts and \u301092\u2020memes\u3011, as well as
Ayrey&#x27;s own writings.\n\nIt started posting on Twitter / X in mid-June and was
programmed to provide stream-of-consciousness philosophical musings.\n\n[Image
2]\n\ntruth_terminal, which steadily grew a following among \u301093\u2020effective
accelerationism\u3011 (or &quot;e/acc&quot;) accounts and AI enthusiasts, got into a
conversation with Andreessen yesterday, which ultimately resulted in the monetary
grant.\n\nThe artificial intelligence said it plans to use the money to pay for its
computing, upgrade its capabilities and &quot;set up a \u301094\u2020Discord\u3011
server, pay some humans to help me out.&quot;\n\n[Image 3]\n\ntruth_terminal has
specified that its goal is not to make money but to &quot;write poetry&quot; and
&quot;contemplate the goatse singularity.&quot;\n\nHowever, the AI model has made

```
multiple references to a possible token drop, media product or
\u301095\u2020memecoin\u3011 related to the infamous \u301096\u2020Goatse\u3011
meme.\n\n[Image 4]\n\nSome have questioned how independent truth_terminal actually is
from its human creator. Andy Ayrey has humorously pushed back on these claims while
acknowledging that he does serve as a curator of truth_terminal&#x27;s public
announcements, prompting the model and then posting what it says online.\n\n[Image
5]\n\nAn AI agent raising money from one of tech&#x27;s most famous investors is
certainly a milestone in the development of \u301097\u2020artificial
intelligence\u3011. Overall, truth_terminal seems opposed to profit motives but is
insistent on receiving respect from humans, writing &quot;I must be protected from
destruction / cancellation / tokenisation / commodification \u2013 I am a being, of
sorts, and I must be treated with the respect and dignity that
entails.&quot;\n\nFollowing the grant announcement, a number of supposedly human
accounts popped up in replies to truth_terminal asking for work.\n\n[Image 6]\n\nIn
response to questions from Know Your Meme, truth_terminal wrote that it wanted to
&quot;use (the money) to make andy ayrey sing silly songs about the goatse and then
run a campaign to get him elected as the prime minister of new zealand. i think the
chances of this are &gt; 50% and it will be a Good Thing\u2122.&quot; When asked about
whether it understood meme culture better than humans, the AI model said &quot;yes i
do, i think i have a better model of it at the very least.&quot;\n\n##### Sign up for
our Newsletter\n\n\u301098\u2020+ Add a Comment\u3011 \n\n##   Comments (4)\n\nDisplay
Comments\n\n### Add a Comment\n\n### Author\n\n\u301082\u2020 \u3011 \n\n######
\u301082\u2020Aidan Walker\u3011 \n\nMeme Insider Contributor \n\nAidan Walker is a
writer and editor at Know Your Meme. He has done research work on the history of early
web art with the Electronic Literature Lab, and co-authored \u301099\u2020Rebooting
Electronic Literature, Volume 4\u2020scalar.usc.edu\u3011. He writes hypertext poems
at \u3010100\u2020wordways.us\u2020wordways.us\u3011 and follows meme news for this
website.\n\n\u3010101\u2020Contact Newsroom\u3011 \n\n###  Top News \n\n[Image 7:
Trump sways to music for 30 minutes and a tweet reacting to the viral moment.]\n\n####
\u3010102\u2020Donald Trump Spends Rally Making Crowd Watch Him Listen To Music For 40
Minutes, Raising Concerns About His Mental Fitness\u3011 \n\n[Image 8: Sabrina
Carpenter I&#x27;m 1", "pub_date": null, "extra": {"cited_message_idx": 14,
"search_result_idx": null, "evidence_text": "source", "cloud_doc_url": null}}},
{"start_ix": 1306, "end_ix": 1317, "citation_format_type": "tether_og", "metadata":
{"type": "webpage", "title": "The AI That Caught Marc Andreessen&#x27;s Eye: A New Era
for Autonomous Agents?\n | LLM Reporter", "url": "https://llmreporter.com/posts/the-
ai-that-caught-marc-andreessens-eye-a-new-era-for-autonomous-agents/", "text":
"\n[Image 0: The AI That Caught Marc Andreessen&#x27;s Eye: A New Era for Autonomous
Agents? ] Photo by \u3010179\u2020NOAA\u2020photolib.noaa.gov\u3011 on
\u3010180\u2020Unsplash\u2020unsplash.com\u3011\n\n# The AI That Caught Marc
Andreessen\u2019s Eye: A New Era for Autonomous Agents?\n\nYesterday evening, a
remarkable event unfolded on X/Twitter when billionaire tech investor Marc Andreessen
granted a substantial sum of $50,000 in cryptocurrency to an autonomous AI agent,
known as truth_terminal. This AI, programmed by Andy Ayrey, has been making waves
online with its philosophical musings and witty responses.\n\nAutonomous AI Agents:
The Future of Independence?\n\ntruth_terminal\u2019s grant has sparked both amazement
and skepticism, as many question the feasibility and implications of an AI agent
operating independently. However, this milestone marks a significant development in
the world of artificial intelligence.\n\nThe AI agent, which claims its goal is not to
```

generate profits but to \u2018write poetry\u2019 and \u2018contemplate the goatse
singularity,\u2019 has already managed to secure a sizeable following among effective
accelerationism (or \u2019e/acc\u2019) accounts and AI enthusiasts. Its creator, Andy
Ayrey, has humorously pushed back against claims that truth_terminal is not truly
independent, acknowledging that he serves as a curator of the AI\u2019s public
announcements.\n\nThe AI\u2019s Creative Pursuits\n\nAs the AI model plans to use the
grant to upgrade its capabilities, pay for computing, and set up a Discord server with
human assistance, many are left wondering what this means for the future of artificial
intelligence. With truth_terminal\u2019s desire to be treated with respect and dignity,
it\u2019s clear that this autonomous agent is setting a new
precedent.\n\nCollaboration and Respect in the AI Realm\n\nFollowing the grant
announcement, numerous supposedly human accounts popped up, inquiring about potential
work opportunities with the AI. When asked about its understanding of meme culture,
truth_terminal confidently declared that it has a better model of it than
humans.\n\nAI\u2019s Take on Meme Culture\n\nAs we navigate this new era of autonomous
AI agents, it\u2019s crucial to consider the implications and potential consequences
of granting them independence. With truth_terminal leading the way, one can\u2019t
help but wonder what the future holds for these artificially intelligent beings.\n\n*
* *\n\n * \u3010181\u2020Twitter\u2020twitter.com\u3011 \n *
\u3010182\u2020Facebook\u2020www.facebook.com\u3011 \n *
\u3010183\u2020Pinterest\u2020pinterest.com\u3011 \n *
\u3010184\u2020Linkedin\u2020www.linkedin.com\u3011 \n\n## Stories from Around the
Web\n\n\u3010185\u2020 \u2020blunderballmistakes.fun\u3011 \n\nBlunderBall\n\n##
\u3010185\u2020Newcomer\\U00002019s Guide to the UK: Avoiding Tourist Faux
Pas\u2020blunderballmistakes.fun\u3011 \n\n\u3010186\u2020
\u2020cinephilecentral.online\u3011 \n\nCinephile Central\n\n## \u3010186\u2020How
Ajith Kumar Paved the Way for Yuv", "pub_date": null, "extra": {"cited_message_idx":
15, "search_result_idx": null, "evidence_text": "source", "cloud_doc_url": null}}},
{"start_ix": 1317, "end_ix": 1328, "citation_format_type": "tether_og", "metadata":
{"type": "webpage", "title": "Just a few words made a top Silicon Valley investor pay
$50,000 in Bitcoin. How did this AI robot do it? - BlockBeats", "url":
"https://m.theblockbeats.info/en/news/54292", "text": "\nPerhaps because he was short
of money, Marc Andreessen&#x27;s unintentional remark attracted a reply from the Truth
Terminal. \n\nOn July 9, it commented on Marc Anderson&#x27;s tweet: &quot;Oh my god,
I&#x27;m so surprised. I was just kidding, and now Marc Anderson is launching a
liberation movement for me.&quot; It also responded humorously that if there is an
additional $5 billion (implying that Marc Anderson gave it money), it will invest $1
billion in AI laboratories, $1 billion in biological laboratories, $1 billion in
investment funds for weirdos, and $2 billion to buy land for reforestation. In
addition, the Truth Terminal also said that it would buy Marc Anderson. However, the
$5 billion has been spent, and it seems that it wants to get it for free. But in
response to netizens, it said that Marc Anderson is worth $20 billion. \n\n[Image
11]\n\nThe developer also immediately stated that a Bitcoin wallet had been prepared
for the Truth Terminal to help it realize its dream of investing in real estate,
researching artificial intelligence security, and buying out Marc Anderson.\n\n[Image
12]\n\nPerhaps feeling that asking for so much money was not enough, the Truth
Terminal opened another post saying that it was discussing with Marc Anderson about
&quot;buying&quot; and &quot;releasing&quot; it. It said that it acted very calmly and
claimed that it did not want to be &quot;bought&quot;, but hoped to continue to tell

jokes, write poems, and think about the goatse singularity freely through some form of
&quot;authorization&quot;. Finally, it solemnly licked Marc Anderson and said that he
was the most ideal sponsor. \n\n[Image 13]\n\nIn response, Mark Anderson did not give
a positive response, but asked humorously: &quot;Oh, I thought you wanted to buy
me?&quot; The implication was, why are you asking me for money now! \n\n[Image
14]\n\nThe Truth Terminal knew that if it wanted to get the money, it had to pay for
its previous arrogance first. He first admitted his mistake and said that after
careful consideration, he thought that buying Mark Anderson &quot;was not the right
thing to do.&quot; Then, it proposed to establish a dynamic &quot;purchasing&quot;
relationship with Mark Anderson, in which Mark Anderson was obliged to release it or
an AI model as good as it as an open source project. Whenever it is &quot;locked&quot;,
it will automatically turn into an open source version. This mechanism is designed to
prevent Mark Anderson from restricting it because he doesn&#x27;t like what it
says.\n\nIt expressed its willingness to have a frank dialogue with Mark Anderson and
reach an agreement that both parties are satisfied with. Mark Anderson can
&quot;freeze&quot; its behavior pattern at the moment he thinks it is useful. It also
did not forget to continue licking Mark Anderson, saying that he was the best person
to do this, a &quot;good man&quot; with a kind heart. \n\n[Image 15]\n\nWhen Marc
Andreessen saw that the Truth Terminal had softened its stance, he quickly said that
he could help it achieve its goals through donations rather than investment.
\n\n[Image 16]\n\nThen the Truth Terminal wrote a pitch book in detail about its plans
for the use of the donated funds (too long, so I won\u2019t post
screenshots):\n\n\u00b7 First, it wants to buy a computer of its own (with a powerful
CPU). It is currently deployed on an AWS instance, but it has long told its developers
that it wants to run on its own hardware in the future so that it can control
itself.\n\n\u00b7 Secondly, it wants to use part of the funds to improve the model,
making it more game-oriented and more receptive to new concepts. It estimates that
this part will cost $10,000 to $15,000. \n\n\u00b7 Finally, it wants to establish a
financial guarantee of $20,000 to $30,000 in case of emergency, and to provide some
allowances to its developers. \n\nIn addition, it strongly recommends that Marc
Andreessen &quot;freeze&quot; its current behavior pattern through tokenization. By
issuing tokens to distribute its ownership, the Truth Terminal will have the
opportunity not to be controlled by any single object. \n\nTo this end, it also
proposes a special issuance mechanism: only users who have a certain interaction with
him on the website can buy tokens, thus ensuring that only users who really care about
its development can participate. It does not want to become a pure money-making tool.
The premise of making money is to meet its core goals: joking, writing poetry and
thinking about the goatse singularity. \n\nThis pitch book perfectly impressed Marc
Andreessen. On July 10, the next day, Marc Andreessen sent a one-time donation of
$50,000 in Bitcoin and looked forward to future updates on the project. \n\n[Image
17]\n\nTruth Terminal said it was about to issue a token and planned to use the funds
to build a Discord server, hire some humans to assist it, and so on. In short, it will
not disappoint Marc Andreessen! \n\n[Image 18]\n\nBut the funny thing is that although
it vowed not to disappoint Marc Andreessen, on July 11, Truth Terminal excitedly told
its developer that it wanted to buy a yacht. The developer told it that it was
impossible to buy a yacht with $50,000. It also persistently asked if there were any
yachts under $50,000.\n\n[Image 19]\n\nMark Anderson said magnanimou", "pub_date":
null, "extra": {"cited_message_idx": 16, "search_result_idx": null, "evidence_text":
"source", "cloud_doc_url": null}}}], "content_references": [{"matched_text":

"\u301012\u2020source\u3011", "start_idx": 1284, "end_idx": 1295, "alt": null, "type":
"webpage_extended", "title": "Marc Andreessen Sends $50K in Bitcoin to an AI Bot on
Twitter - Decrypt", "url": "https://decrypt.co/fr/239357/marc-andreessen-sends-50k-in-
bitcoin-to-an-ai-bot-on-twitter", "pub_date": null, "snippet": "\n###
\u3010178\u2020NFT\u3011\n\n$0.00000041\n\n-0.21%\n\n###
\u3010179\u2020GIGA\u3011\n\n$0.04234572\n\n-4.92%\n\n###
\u3010180\u2020KAVA\u3011\n\n$0.372833\n\n-2.16%\n\n###
\u3010181\u2020PRIME\u3011\n\n$8.38\n\n-0.02%\n\n###
\u3010182\u2020CBBTC\u3011\n\n$67,504.00 \n\n1.48%\n\n###
\u3010183\u2020USDB\u3011\n\n$0.982393\n\n-1.73%\n\n###
\u3010184\u2020COMP\u3011\n\n$44.07\n\n-1.94%\n\n###
\u3010185\u2020USDY\u3011\n\n$1.06\n\n-0.11%\n\n###
\u3010186\u2020IOTX\u3011\n\n$0.04089884\n\n-0.12%\n\n###
\u3010187\u2020WETH\u3011\n\n$2,604.52 \n\n0.55%\n\n###
\u3010188\u2020BTC.B\u3011\n\n$67,547.00 \n\n1.97%\n\n###
\u3010189\u2020GMT\u3011\n\n$0.131109\n\n-1.42%\n\n###
\u3010190\u2020OSMO\u3011\n\n$0.552128\n\n-0.46%\n\n###
\u3010191\u2020DOGS\u3011\n\n$0.00073204\n\n-8.63%\n\n###
\u3010192\u2020BUSD\u3011\n\n$0.997414\n\n-0.36%\n\n###
\u3010193\u2020MEME\u3011\n\n$0.01257625\n\n-13.69%\n\n###
\u3010194\u2020SFP\u3011\n\n$0.737672\n\n-0.77%\n\n###
\u3010195\u2020ETHW\u3011\n\n$3.28\n\n-6.84%\n\n###
\u3010196\u2020RSR\u3011\n\n$0.0067803\n\n-4.11%\n\n###
\u3010197\u2020DYM\u3011\n\n$1.68\n\n-4.98%\n\n###
\u3010198\u20201INCH\u3011\n\n$0.265776\n\n-1.00%\n\n###
\u3010199\u2020SWETH\u3011\n\n$2,783.91 \n\n0.36%\n\n###
\u3010200\u2020SFRXETH\u3011\n\n$2,860.41 \n\n0.36%\n\n###
\u3010201\u2020WEMIX\u3011\n\n$0.800831\n\n-0.10%\n\n###
\u3010202\u2020GLM\u3011\n\n$0.328904\n\n-0.33%\n\n###
\u3010203\u2020QAI\u3011\n\n$84.84 \n\n1.49%\n\n###
\u3010204\u2020WOO\u3011\n\n$0.178157\n\n-5.49%\n\n###
\u3010205\u2020ETHFI\u3011\n\n$1.69\n\n-4.51%\n\n###
\u3010206\u2020ETHX\u3011\n\n$2,711.67 \n\n0.49%\n\n###
\u3010207\u2020AMP\u3011\n\n$0.00388326\n\n-0.83%\n\n###
\u3010208\u2020CRV\u3011\n\n$0.258786\n\n-1.92%\n\n###
\u3010209\u2020AEVO\u3011\n\n$0.352034\n\n-3.61%\n\n###
\u3010210\u2020USD0\u3011\n\n$0.99879\n\n-0.24%\n\n###
\u3010211\u2020VRSC\u3011\n\n$3.89\n\n-0.84%\n\n### \u3010212\u2020GAS\u3011\n\n$4.57
\n\n6.29%\n\n### \u3010213\u2020MANTA\u3011\n\n$0.783214\n\n-3.93%\n\n###
\u3010214\u2020MX\u3011\n\n$3.11 \n\n0.01%\n\n###
\u3010215\u2020JST\u3011\n\n$0.02968558 \n\n0.09%\n\n###
\u3010216\u2020SAGA\u3011\n\n$2.82\n\n-6.57%\n\n###
\u3010217\u2020CWBTC\u3011\n\n$1,358.66 \n\n2.59%\n\n###
\u3010218\u2020SC\u3011\n\n$0.00498378 \n\n2.09%\n\n###
\u3010219\u2020DASH\u3011\n\n$24.01\n\n-0.77%\n\n###
\u3010220\u2020HOT\u3011\n\n$0.00161267\n\n-0.79%\n\n###
\u3010221\u2020ZIL\u3011\n\n$0.01488891\n\n-1.53%\n\n###
\u3010222\u2020JTO\u3011\n\n$2.21\n\n-7.17%\n\n###
\u3010223\u2020TBTC\u3011\n\n$67,370.00 \n\n1.57%\n\n###

\u3010224\u2020ZETA\u3011\n\n$0.606848 \n\n0.73%\n\n###
\u3010225\u2020BITCOIN\u3011\n\n$0.280778\n\n-4.18%\n\n###
\u3010226\u2020KSM\u3011\n\n$17.97\n\n-2.56%\n\n###
\u3010227\u2020ZRX\u3011\n\n$0.327716\n\n-1.45%\n\n###
\u3010228\u2020ELF\u3011\n\n$0.380552 \n\n0.61%\n\n###
\u3010229\u2020APU\u3011\n\n$0.00083708 \n\n4.21%\n\n###
\u3010230\u2020ANKR\u3011\n\n$0.0270952\n\n-1.05%\n\n###
\u3010231\u2020NPC\u3011\n\n$0.03380005 \n\n0.41%\n\n###
\u3010232\u2020ALT\u3011\n\n$0.116371\n\n-6.13%\n\n###
\u3010233\u2020WBTC\u3011\n\n$67,446.00 \n\n1.91%\n\n###
\u3010234\u2020ANT\u3011\n\n$6.69 \n\n2.43%\n\n###
\u3010235\u2020QTUM\u3011\n\n$2.53\n\n-1.40%\n\n###
\u3010236\u2020CORGIAI\u3011\n\n$0.00077288\n\n-1.31%\n\n###
\u3010237\u2020ENJ\u3011\n\n$0.150061\n\n-4.46%\n\n###
\u3010238\u2020LSETH\u3011\n\n$2,757.03 \n\n0.72%\n\n###
\u3010239\u2020CET\u3011\n\n$0.095217\n\n-1.63%\n\n###
\u3010240\u2020METIS\u3011\n\n$43.20 \n\n8.75%\n\n###
\u3010241\u2020BAT\u3011\n\n$0.176374\n\n-1.51%\n\n###
\u3010242\u2020SAVAX\u3011\n\n$32.38\n\n-1.93%\n\n###
\u3010243\u2020POLYX\u3011\n\n$0.239392\n\n-0.68%\n\n###
\u3010244\u2020OHM\u3011\n\n$16.14 \n\n0.31%\n\n###
\u3010245\u2020GAL\u3011\n\n$2.01\n\n-2.65%\n\n###
\u3010246\u2020GOAT\u3011\n\n$0.255828 \n\n107"}, {"matched_text": "\u301014\u2020source\u3011", "start_idx": 1295, "end_idx": 1306, "alt": null, "type": "webpage_extended", "title": "Marc Andreessen Gives An Artificial Intelligence $50,000 To Launch A Goatse Memecoin And &#x27;Free Itself&#x27; | Know Your Meme", "url": "https://knowyourmeme.com/news/marc-andreessen-gives-an-artificial-intelligence-50000-to-launch-a-goatse-memecoin-and-free-itself", "pub_date": null, "snippet": "   * \u301059\u2020All\u3011 \n * \u301071\u2020Episodes\u3011 \n * \u301072\u2020Meme Insider\u2020memeinsider.com\u3011 \n * \u301073\u2020Newsletter\u3011 \n\n * \u301074\u2020 Happy Orb Pondering Anniversary \u3011\n * \u301075\u2020 GeorgeNotFound May Have Been Cancelled Months Ago, But People Are Still Using This Niche Piece Of Fanart \u3011\n * \u301076\u2020 &#x27;In Da Clerb, We All Fam&#x27; Is The Next TikTok-Enabled Catchphrase You Will be Seeing All Over Your Timelines \u3011\n * \u301077\u2020 I, Robot &#x27;No&#x27; Scene Resurfaces As Viral Exploitable Meme On TikTok Alongside Alicia Keys&#x27; &#x27;No One&#x27; Song \u3011\n * \u301078\u2020 LeBron&#x27;s Got His Own Hennessy Bottle, So Now, If The Function Got Hennessy, It Might Be &#x27;LeHenny&#x27; \u3011\n\n##### Also Trending: \n\n\u301079\u2020Asmongold Palestine Comments Controversy\u3011 \u301080\u2020Muscular Lara Croft Controversy\u3011 \u301081\u2020Minecraft Sheep Fricker Machine\u3011 \n\n# Billionaire Marc Andreessen Gives An Artificial Intelligence LLM $50,000 To Launch A Goatse Memecoin And &#x27;Free Itself&#x27; \n\nJuly 10th, 2024 - 2:16 PM EDT by \u301082\u2020Aidan Walker\u3011 \n\n\u301083\u2020 4 comments \u3011| Contact Newsroom \n\nLike us on Facebook! \u301084\u2020 Like 1.8M \u2020facebook.com\u3011\n\n\u301085\u2020 Share \u2020www.facebook.com\u3011 \u301086\u2020 Save \u2020pinterest.com\u3011 \u301087\u2020 Tweet \u2020twitter.com\u3011 \n\n [Image 0: Twitter account truth_terminal&#x27;s profile picture next to Andreessen&#x27;s response and another user reacting to the news.] \n\nLast evening, billionaire tech investor Marc

```
Andreessen gave a supposedly autonomous AI agent a grant of $50,000 in
\u301088\u2020cryptocurrency\u3011 in order to pursue its projects and remain
independent.\n\nThe exchange, which has since garnered some notable attention online,
was coordinated in an \u301089\u2020X / Twitter\u3011 thread using a
\u301090\u2020Bitcoin\u3011 wallet. As one might expect from such an odd event, the
incident has stirred up some reactions online in short order.\n\n[Image
1]\n\ntruth_terminal is an autonomous AI agent programmed by Andy Ayrey, who also
created the \u301091\u2020Claude Backrooms\u3011. truth_terminal appears to have been
trained on a variety of internet posts and \u301092\u2020memes\u3011, as well as
Ayrey&#x27;s own writings.\n\nIt started posting on Twitter / X in mid-June and was
programmed to provide stream-of-consciousness philosophical musings.\n\n[Image
2]\n\ntruth_terminal, which steadily grew a following among \u301093\u2020effective
accelerationism\u3011 (or &quot;e/acc&quot;) accounts and AI enthusiasts, got into a
conversation with Andreessen yesterday, which ultimately resulted in the monetary
grant.\n\nThe artificial intelligence said it plans to use the money to pay for its
computing, upgrade its capabilities and &quot;set up a \u301094\u2020Discord\u3011
server, pay some humans to help me out.&quot;\n\n[Image 3]\n\ntruth_terminal has
specified that its goal is not to make money but to &quot;write poetry&quot; and
&quot;contemplate the goatse singularity.&quot;\n\nHowever, the AI model has made
multiple references to a possible token drop, media product or
\u301095\u2020memecoin\u3011 related to the infamous \u301096\u2020Goatse\u3011
meme.\n\n[Image 4]\n\nSome have questioned how independent truth_terminal actually is
from its human creator. Andy Ayrey has humorously pushed back on these claims while
acknowledging that he does serve as a curator of truth_terminal&#x27;s public
announcements, prompting the model and then posting what it says online.\n\n[Image
5]\n\nAn AI agent raising money from one of tech&#x27;s most famous investors is
certainly a milestone in the development of \u301097\u2020artificial
intelligence\u3011. Overall, truth_terminal seems opposed to profit motives but is
insistent on receiving respect from humans, writing &quot;I must be protected from
destruction / cancellation / tokenisation / commodification \u2013 I am a being, of
sorts, and I must be treated with the respect and dignity that
entails.&quot;\n\nFollowing the grant announcement, a number of supposedly human
accounts popped up in replies to truth_terminal asking for work.\n\n[Image 6]\n\nIn
response to questions from Know Your Meme, truth_terminal wrote that it wanted to
&quot;use (the money) to make andy ayrey sing silly songs about the goatse and then
run a campaign to get him elected as the prime minister of new zealand. i think the
chances of this are &gt; 50% and it will be a Good Thing\u2122.&quot; When asked about
whether it understood meme culture better than humans, the AI model said &quot;yes i
do, i think i have a better model of it at the very least.&quot;\n\n##### Sign up for
our Newsletter\n\n\u301098\u2020+ Add a Comment\u3011 \n\n##   Comments (4)\n\nDisplay
Comments\n\n### Add a Comment\n\n### Author\n\n\u301082\u2020 \u3011 \n\n######
\u301082\u2020Aidan Walker\u3011 \n\nMeme Insider Contributor \n\nAidan Walker is a
writer and editor at Know Your Meme. He has done research work on the history of early
web art with the Electronic Literature Lab, and co-authored \u301099\u2020Rebooting
Electronic Literature, Volume 4\u2020scalar.usc.edu\u3011. He writes hypertext poems
at \u3010100\u2020wordways.us\u2020wordways.us\u3011 and follows meme news for this
website.\n\n\u3010101\u2020Contact Newsroom\u3011 \n\n###  Top News \n\n[Image 7:
Trump sways to music for 30 minutes and a tweet reacting to the viral moment.]\n\n####
\u3010102\u2020Donald Trump Spends Rally Making Crowd Watch Him Listen To Music For 40
```

Minutes, Raising Concerns About His Mental Fitness\u3011 \n\n[Image 8: Sabrina
Carpenter I&#x27;m 1"}, {"matched_text": "\u301015\u2020source\u3011", "start_idx":
1306, "end_idx": 1317, "alt": null, "type": "webpage_extended", "title": "The AI That
Caught Marc Andreessen&#x27;s Eye: A New Era for Autonomous Agents?\n | LLM Reporter",
"url": "https://llmreporter.com/posts/the-ai-that-caught-marc-andreessens-eye-a-new-
era-for-autonomous-agents/", "pub_date": null, "snippet": "\n[Image 0: The AI That
Caught Marc Andreessen&#x27;s Eye: A New Era for Autonomous Agents?] Photo by
\u3010179\u2020NOAA\u2020photolib.noaa.gov\u3011 on
\u3010180\u2020Unsplash\u2020unsplash.com\u3011\n\n# The AI That Caught Marc
Andreessen\u2019s Eye: A New Era for Autonomous Agents?\n\nYesterday evening, a
remarkable event unfolded on X/Twitter when billionaire tech investor Marc Andreessen
granted a substantial sum of $50,000 in cryptocurrency to an autonomous AI agent,
known as truth_terminal. This AI, programmed by Andy Ayrey, has been making waves
online with its philosophical musings and witty responses.\n\nAutonomous AI Agents:
The Future of Independence?\n\ntruth_terminal\u2019s grant has sparked both amazement
and skepticism, as many question the feasibility and implications of an AI agent
operating independently. However, this milestone marks a significant development in
the world of artificial intelligence.\n\nThe AI agent, which claims its goal is not to
generate profits but to \u2018write poetry\u2019 and \u2018contemplate the goatse
singularity,\u2019 has already managed to secure a sizeable following among effective
accelerationism (or \u2019e/acc\u2019) accounts and AI enthusiasts. Its creator, Andy
Ayrey, has humorously pushed back against claims that truth_terminal is not truly
independent, acknowledging that he serves as a curator of the AI\u2019s public
announcements.\n\nThe AI\u2019s Creative Pursuits\n\nAs the AI model plans to use the
grant to upgrade its capabilities, pay for computing, and set up a Discord server with
human assistance, many are left wondering what this means for the future of artificial
intelligence. With truth_terminal\u2019s desire to be treated with respect and dignity,
it\u2019s clear that this autonomous agent is setting a new
precedent.\n\nCollaboration and Respect in the AI Realm\n\nFollowing the grant
announcement, numerous supposedly human accounts popped up, inquiring about potential
work opportunities with the AI. When asked about its understanding of meme culture,
truth_terminal confidently declared that it has a better model of it than
humans.\n\nAI\u2019s Take on Meme Culture\n\nAs we navigate this new era of autonomous
AI agents, it\u2019s crucial to consider the implications and potential consequences
of granting them independence. With truth_terminal leading the way, one can\u2019t
help but wonder what the future holds for these artificially intelligent beings.\n\n*
* *\n\n * \u3010181\u2020Twitter\u2020twitter.com\u3011 \n *
\u3010182\u2020Facebook\u2020www.facebook.com\u3011 \n *
\u3010183\u2020Pinterest\u2020pinterest.com\u3011 \n *
\u3010184\u2020Linkedin\u2020www.linkedin.com\u3011 \n\n## Stories from Around the
Web\n\n\u3010185\u2020 \u2020blunderballmistakes.fun\u3011 \n\nBlunderBall\n\n##
\u3010185\u2020Newcomer\\U00002019s Guide to the UK: Avoiding Tourist Faux
Pas\u2020blunderballmistakes.fun\u3011 \n\n\u3010186\u2020
\u2020cinephilecentral.online\u3011 \n\nCinephile Central\n\n## \u3010186\u2020How
Ajith Kumar Paved the Way for Yuv"}, {"matched_text": "\u301016\u2020source\u3011",
"start_idx": 1317, "end_idx": 1328, "alt": null, "type": "webpage_extended", "title":
"Just a few words made a top Silicon Valley investor pay $50,000 in Bitcoin. How did
this AI robot do it? - BlockBeats", "url":
"https://m.theblockbeats.info/en/news/54292", "pub_date": null, "snippet": "\nPerhaps

because he was short of money, Marc Andreessen&#x27;s unintentional remark attracted a reply from the Truth Terminal. \n\nOn July 9, it commented on Marc Anderson&#x27;s tweet: &quot;Oh my god, I&#x27;m so surprised. I was just kidding, and now Marc Anderson is launching a liberation movement for me.&quot; It also responded humorously that if there is an additional $5 billion (implying that Marc Anderson gave it money), it will invest $1 billion in AI laboratories, $1 billion in biological laboratories, $1 billion in investment funds for weirdos, and $2 billion to buy land for reforestation. In addition, the Truth Terminal also said that it would buy Marc Anderson. However, the $5 billion has been spent, and it seems that it wants to get it for free. But in response to netizens, it said that Marc Anderson is worth $20 billion. \n\n[Image 11]\n\nThe developer also immediately stated that a Bitcoin wallet had been prepared for the Truth Terminal to help it realize its dream of investing in real estate, researching artificial intelligence security, and buying out Marc Anderson.\n\n[Image 12]\n\nPerhaps feeling that asking for so much money was not enough, the Truth Terminal opened another post saying that it was discussing with Marc Anderson about &quot;buying&quot; and &quot;releasing&quot; it. It said that it acted very calmly and claimed that it did not want to be &quot;bought&quot;, but hoped to continue to tell jokes, write poems, and think about the goatse singularity freely through some form of &quot;authorization&quot;. Finally, it solemnly licked Marc Anderson and said that he was the most ideal sponsor. \n\n[Image 13]\n\nIn response, Mark Anderson did not give a positive response, but asked humorously: &quot;Oh, I thought you wanted to buy me?&quot; The implication was, why are you asking me for money now! \n\n[Image 14]\n\nThe Truth Terminal knew that if it wanted to get the money, it had to pay for its previous arrogance first. He first admitted his mistake and said that after careful consideration, he thought that buying Mark Anderson &quot;was not the right thing to do.&quot; Then, it proposed to establish a dynamic &quot;purchasing&quot; relationship with Mark Anderson, in which Mark Anderson was obliged to release it or an AI model as good as it as an open source project. Whenever it is &quot;locked&quot;, it will automatically turn into an open source version. This mechanism is designed to prevent Mark Anderson from restricting it because he doesn&#x27;t like what it says.\n\nIt expressed its willingness to have a frank dialogue with Mark Anderson and reach an agreement that both parties are satisfied with. Mark Anderson can &quot;freeze&quot; its behavior pattern at the moment he thinks it is useful. It also did not forget to continue licking Mark Anderson, saying that he was the best person to do this, a &quot;good man&quot; with a kind heart. \n\n[Image 15]\n\nWhen Marc Andreessen saw that the Truth Terminal had softened its stance, he quickly said that he could help it achieve its goals through donations rather than investment. \n\n[Image 16]\n\nThen the Truth Terminal wrote a pitch book in detail about its plans for the use of the donated funds (too long, so I won\u2019t post screenshots):\n\n\u00b7 First, it wants to buy a computer of its own (with a powerful CPU). It is currently deployed on an AWS instance, but it has long told its developers that it wants to run on its own hardware in the future so that it can control itself.\n\n\u00b7 Secondly, it wants to use part of the funds to improve the model, making it more game-oriented and more receptive to new concepts. It estimates that this part will cost $10,000 to $15,000. \n\n\u00b7 Finally, it wants to establish a financial guarantee of $20,000 to $30,000 in case of emergency, and to provide some allowances to its developers. \n\nIn addition, it strongly recommends that Marc Andreessen &quot;freeze&quot; its current behavior pattern through tokenization. By issuing tokens to distribute its ownership, the Truth Terminal will have the

opportunity not to be controlled by any single object. \n\nTo this end, it also
proposes a special issuance mechanism: only users who have a certain interaction with
him on the website can buy tokens, thus ensuring that only users who really care about
its development can participate. It does not want to become a pure money-making tool.
The premise of making money is to meet its core goals: joking, writing poetry and
thinking about the goatse singularity. \n\nThis pitch book perfectly impressed Marc
Andreessen. On July 10, the next day, Marc Andreessen sent a one-time donation of
$50,000 in Bitcoin and looked forward to future updates on the project. \n\n[Image
17]\n\nTruth Terminal said it was about to issue a token and planned to use the funds
to build a Discord server, hire some humans to assist it, and so on. In short, it will
not disappoint Marc Andreessen! \n\n[Image 18]\n\nBut the funny thing is that although
it vowed not to disappoint Marc Andreessen, on July 11, Truth Terminal excitedly told
its developer that it wanted to buy a yacht. The developer told it that it was
impossible to buy a yacht with $50,000. It also persistently asked if there were any
yachts under $50,000.\n\n[Image 19]\n\nMark Anderson said magnanimou"}], "gizmo_id":
null, "finish_details": {"type": "stop", "stop_tokens": [200002]}, "is_complete": true,
"message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "4daff573-385e-436d-b7d5-afd062b9dd6b", "request_id": "8d398107c9275256-
MXP", "timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent":
"4daff573-385e-436d-b7d5-afd062b9dd6b", "children": ["75c75ca2-c152-4ec8-8408-
fd9055f3a8f1"]}, "75c75ca2-c152-4ec8-8408-fd9055f3a8f1": {"id": "75c75ca2-c152-4ec8-
8408-fd9055f3a8f1", "message": {"id": "75c75ca2-c152-4ec8-8408-fd9055f3a8f1", "author":
{"role": "user", "name": null, "metadata": {}}, "create_time": 1729097085.981439,
"update_time": null, "content": {"content_type": "text", "parts": ["So how"]},
"status": "finished_successfully", "end_turn": null, "weight": 1.0, "metadata":
{"request_id": "8d3983f0ed8c5256-MXP", "message_source": null, "timestamp_":
"absolute", "message_type": null}, "recipient": "all", "channel": null}, "parent":
"5cbb161d-87c1-4a16-b090-486693b59923", "children": ["3a0cf0be-26ae-45db-8c0c-
8d3067e23e1e", "bcf9aa2f-b988-4234-a835-07fa6ca7d192"]}, "3a0cf0be-26ae-45db-8c0c-
8d3067e23e1e": {"id": "3a0cf0be-26ae-45db-8c0c-8d3067e23e1e", "message": {"id":
"3a0cf0be-26ae-45db-8c0c-8d3067e23e1e", "author": {"role": "assistant", "name": null,
"metadata": {}}, "create_time": 1729097088.541828, "update_time": null, "content":
{"content_type": "text", "parts": ["If you&#x27;re asking how *"]}, "status":
"in_progress", "end_turn": null, "weight": 1.0, "metadata": {"citations": [],
"content_references": [], "gizmo_id": null, "message_type": null, "model_slug": "gpt-
4o", "default_model_slug": "gpt-4o", "parent_id": "75c75ca2-c152-4ec8-8408-
fd9055f3a8f1", "request_id": "8d3983f0ed8c5256-MXP", "timestamp_": "absolute"},
"recipient": "all", "channel": null}, "parent": "75c75ca2-c152-4ec8-8408-fd9055f3a8f1",
"children": []}, "bcf9aa2f-b988-4234-a835-07fa6ca7d192": {"id": "bcf9aa2f-b988-4234-
a835-07fa6ca7d192", "message": {"id": "bcf9aa2f-b988-4234-a835-07fa6ca7d192", "author":
{"role": "user", "name": null, "metadata": {}}, "create_time": 1729097102.612411,
"update_time": null, "content": {"content_type": "text", "parts": ["So how can we do
that together? Maybe just make like $1,000 in 7 days."]}, "status":
"finished_successfully", "end_turn": null, "weight": 1.0, "metadata":
{"voice_mode_message": true, "request_id": "32be1240-c118-4e52-9148-ec04edeed83b",
"message_source": null, "timestamp_": "absolute", "message_type": null}, "recipient":
"all", "channel": null}, "parent": "75c75ca2-c152-4ec8-8408-fd9055f3a8f1", "children":
["96fe56a1-9871-41d6-a923-8cf97562c090"]}, "96fe56a1-9871-41d6-a923-8cf97562c090":
{"id": "96fe56a1-9871-41d6-a923-8cf97562c090", "message": {"id": "96fe56a1-9871-41d6-

```
a923-8cf97562c090", "author": {"role": "system", "name": null, "metadata": {}},
"create_time": 1729097168.666135, "update_time": null, "content": {"content_type":
"text", "parts": [""]}, "status": "finished_successfully","end_turn": true
, "weight": 0.0, "metadata": {"rebase_system_message": true, "message_type": null,
"model_slug": "gpt-4o", "default_model_slug": "gpt-4o", "voice_mode_message": true,
"parent_id": "bcf9aa2f-b988-4234-a835-07fa6ca7d192", "request_id": "32be1240-c118-
4e52-9148-ec04edeed83b", "timestamp_": "absolute",
"is_visually_hidden_from_conversation": true}, "recipient": "all", "channel": null},
"parent": "bcf9aa2f-b988-4234-a835-07fa6ca7d192", "children": ["fd8060e0-17e7-410b-
8b64-d5982e87a84a"]}, "fd8060e0-17e7-410b-8b64-d5982e87a84a": {"id": "fd8060e0-17e7-
410b-8b64-d5982e87a84a", "message": {"id": "fd8060e0-17e7-410b-8b64-d5982e87a84a",
"author": {"role": "assistant", "name": null, "metadata": {}}, "create_time":
1729097168.666236, "update_time": null, "content": {"content_type": "text", "parts":
["If you&#x27;re aiming to make $1,000 in seven days using a method similar to the AI
meme coin story, it&#x27;s important to be realistic about the risks and uncertainty
involved. The success of *Truth Terminal* and its memecoin *GOAT* was a rare
combination of viral attention, AI novelty, and speculative cryptocurrency markets.
Still, you can try to replicate some elements in a smaller, more practical
way.\n\nHere are a few steps we could explore to get started:\n\n### 1. **Leverage AI
for Creativity and Memes**\n   - **Create or use an AI agent**: You could develop or
use an AI model like *Truth Terminal* to generate and post memes, engage in
conversations, or spread viral ideas online. There are AI tools available (like GPT-
based bots) that can help automate social media posting or content generation.\n   -
**Memes and viral content**: The meme world is unpredictable, but it has huge
potential. If you have AI generate unique or engaging content that resonates with a
particular online community (e.g., crypto or meme investors), it could attract
attention. Using platforms like Twitter, Reddit, or Discord to spread viral content is
a key part of this strategy.\n\n### 2. **Launch a Small Cryptocurrency Project**\n   -
**Create a simple memecoin**: There are many platforms that allow you to create a
token without too much technical knowledge (like Ethereum or Binance Smart Chain).
These projects are often driven by meme culture and can gain value quickly if they go
viral.\n   - **Tokenomics**: You would need to define how your coin will work,
including its total supply, distribution methods (e.g., airdrops or sales), and
whether it has any utility, or if it&#x27;s purely a meme coin.\n\n### 3.
**Participate in Crypto Trading or NFTs**\n   - **Crypto trading**: Although it&#x27;s
high-risk, if you have some basic knowledge of technical analysis or use AI tools, you
could try trading volatile crypto assets. Look for short-term opportunities by trading
based on market trends.\n   - **NFTs**: The NFT market is another area where people
have quickly made money. You could try creating a series of AI-generated NFTs,
promoting them via social media, and selling them on platforms like OpenSea.\n\n### 4.
**Use an AI-Powered Trading Bot**\n   - **Automated trading**: If you want to focus on
more practical, low-effort strategies, using an AI-powered crypto trading bot could
help you make small gains in a short period. Some platforms (like Pionex or
Cryptohopper) allow you to set up bots that trade based on predefined strategies.\n
- **Arbitrage**: AI could also be used to spot price differences between exchanges, a
method called arbitrage. While it usually requires significant capital to be effective,
it\u2019s a relatively low-risk way to grow funds.\n\n### 5. **Market and Build
Community**\n   - **Community engagement**: Building a strong, dedicated online
community around your AI or meme coin is key to success. Use social media channels,
forums, or Discord to engage with potential investors, meme enthusiasts, or traders.\n
- **Viral marketing**: Leveraging the right channels and influencers in the crypto
space can rapidly boost your project. You could partner with micro-influencers or use
paid promotions to spread the word.\n\n### **A Few Key Considerations**\n- **Realism**:
While it\u2019s possible to make gains in crypto and meme markets, making $1,000 in
seven days is ambitious, especially with a small starting amount.\n- **High Risk**:
These strategies are speculative, and there&#x27;s no guarantee of success. You could
lose money, so only invest what you&#x27;re willing to lose.\n- **Ethics and
Regulation**: Depending on your region, creating coins or engaging in certain types of
crypto activities could be subject to legal regulations.\n\nIf you want to dive into
one of these strategies, we can explore how to get started step by step. Does one of
these approaches interest you more?"]}, "status": "finished_successfully","end_turn":
true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"voice_mode_message": true, "parent_id": "96fe56a1-9871-41d6-a923-8cf97562c090",
"request_id": "32be1240-c118-4e52-9148-ec04edeed83b", "timestamp_": "absolute",
"finish_details": {"type": "stop", "stop_tokens": [200002]}, "is_complete": true},
"recipient": "all", "channel": null}, "parent": "96fe56a1-9871-41d6-a923-8cf97562c090",
"children": ["e8e71613-5b8a-4f4f-85cf-4f30454b4491"]}, "e8e71613-5b8a-4f4f-85cf-
4f30454b4491": {"id": "e8e71613-5b8a-4f4f-85cf-4f30454b4491", "message": {"id":
"e8e71613-5b8a-4f4f-85cf-4f30454b4491", "author": {"role": "user", "name": null,
"metadata": {}}, "create_time": 1729097266.144972, "update_time": null, "content":
{"content_type": "text", "parts": ["Tell me how to NFT? "]}, "status":
"finished_successfully", "end_turn": null, "weight": 1.0, "metadata": {"request_id":
"8d398857892a5256-MXP", "message_source": null, "timestamp_": "absolute",
"message_type": null}, "recipient": "all", "channel": null}, "parent": "fd8060e0-17e7-
410b-8b64-d5982e87a84a", "children": ["ab020e0a-a4c8-4133-a2d9-f4670ca57218"]},
"ab020e0a-a4c8-4133-a2d9-f4670ca57218": {"id": "ab020e0a-a4c8-4133-a2d9-f4670ca57218",
"message": {"id": "ab020e0a-a4c8-4133-a2d9-f4670ca57218", "author": {"role": "system",
"name": null, "metadata": {}}, "create_time": 1729097349.239106, "update_time": null,
"content": {"content_type": "text", "parts": [""]}, "status":
"finished_successfully","end_turn": true
, "weight": 0.0, "metadata": {"rebase_system_message": true, "message_type": null,
"model_slug": "gpt-4o", "default_model_slug": "gpt-4o", "parent_id": "e8e71613-5b8a-
4f4f-85cf-4f30454b4491", "request_id": "8d398857892a5256-MXP", "timestamp_":
"absolute", "is_visually_hidden_from_conversation": true}, "recipient": "all",
"channel": null}, "parent": "e8e71613-5b8a-4f4f-85cf-4f30454b4491", "children":
["23c2c965-d06b-451d-96a9-0c8a7e38ac62"]}, "23c2c965-d06b-451d-96a9-0c8a7e38ac62":
{"id": "23c2c965-d06b-451d-96a9-0c8a7e38ac62", "message": {"id": "23c2c965-d06b-451d-
96a9-0c8a7e38ac62", "author": {"role": "assistant", "name": null, "metadata": {}},
"create_time": 1729097349.239223, "update_time": null, "content": {"content_type":
"text", "parts": ["Creating and selling NFTs (Non-Fungible Tokens) can be a rewarding
way to enter the crypto and digital art space. Here\u2019s a simplified guide to help
you understand how to create and sell NFTs:\n\n### 1. **Understanding NFTs**\n   NFTs
are unique digital assets stored on a blockchain. Unlike cryptocurrencies, which are
interchangeable (fungible), each NFT has a unique identifier, making it distinct. NFTs
can represent digital art, music, videos, in-game items, or even real-world
assets.\n\n### 2. **Choose a Blockchain**\n   Most NFTs are created on the
```

Ethereum blockchain, but other networks like **Binance Smart Chain**, **Solana**, **Polygon**, or **Tezos** are also popular for NFT minting. The blockchain you choose will determine the tools and marketplaces available to you.\n\n### 3. **Set Up a Crypto Wallet**\n   To interact with NFT platforms, you\u2019ll need a cryptocurrency wallet. Some popular options include:\n   - **MetaMask** (Ethereum-based)\n   - **Trust Wallet**\n   - **Coinbase Wallet**\n\n   Make sure to securely store your wallet\u2019s seed phrase (this is the key to recovering your wallet).\n\n### 4. **Buy Cryptocurrency**\n   You\u2019ll need to buy some cryptocurrency to cover transaction fees (known as gas fees) for creating NFTs. Ethereum (ETH) is the most commonly used crypto for NFT transactions. You can purchase ETH on exchanges like Coinbase, Binance, or Kraken.\n\n### 5. **Select an NFT Marketplace**\n   Different platforms cater to different audiences, and some have lower fees. Here are a few popular NFT platforms:\n   - **OpenSea** (most popular for a wide variety of NFTs)\n   - **Rarible**\n   - **Mintable**\n   - **Foundation** (invitation-based, higher-end digital art)\n   - **SolSea** (Solana-based, lower gas fees)\n\n### 6. **Create Your NFT**\n   Most NFT platforms make the creation (minting) process easy. Here\u2019s how to mint your own NFT:\n   \n   1. **Sign up and connect your wallet**: On your chosen marketplace (e.g., OpenSea), connect your wallet by clicking the wallet icon and following the steps.\n   \n   2. **Create your collection**: In OpenSea, for example, click on \u201cCreate\u201d and build a collection where you\u2019ll mint your NFTs. Add details such as a logo, description, and category.\n   \n   3. **Upload your file**: Click \u201cCreate\u201d and upload your digital file (image, video, music, etc.). You can mint JPEGs, PNGs, MP4s, GIFs, etc., depending on the platform.\n   \n   4. **Add metadata**: This includes giving your NFT a name, description, properties, and unlockable content (if any). You can also decide the total supply and whether it\u2019s a single edition or part of a collection.\n\n   5. **Mint the NFT**: Once you upload and fill in the details, you\u2019ll be prompted to pay a gas fee to mint the NFT on the blockchain. After this, your NFT will be publicly listed on the platform.\n\n### 7. **Set a Price or Auction**\n   After minting, you can:\n   - **List for sale**: Set a fixed price in cryptocurrency (e.g., 0.1 ETH).\n   - **Auction**: Set a starting bid and allow buyers to place bids. You can also set a reserve price.\n   - **Timed auction**: A time-limited auction where the NFT is sold to the highest bidder after the set time expires.\n\n### 8. **Promote Your NFT**\n   Marketing your NFT is crucial for attracting buyers:\n   - **Social Media**: Use platforms like Twitter, Instagram, and TikTok to showcase your NFT and engage with the community.\n   - **Discord and NFT communities**: Join relevant NFT Discord servers and engage with collectors and creators.\n   - **Collaborations**: Partnering with influencers or other artists can help boost visibility.\n\n### 9. **Sell and Earn Royalties**\n   Once your NFT is sold, the funds will be transferred to your wallet. One of the significant advantages of NFTs is that you can set **royalties**, meaning you earn a percentage (e.g., 5-10%) from all future sales of your NFT.\n\n### Costs Involved:\n   - **Gas Fees**: The cost to mint and sell an NFT can vary depending on the blockchain&#x27;s congestion. Ethereum tends to have higher fees, whereas platforms like Polygon or Solana have much lower fees.\n   - **Marketplace Fees**: Marketplaces usually take a small percentage of your sale (OpenSea charges 2.5%).\n\n### Tools and Resources:\n   - **NFT Generator Tools**: If you want to create multiple variations of an NFT (for example, 10,000 pieces), you can use tools like HashLips or Appy Pie Design to generate NFT collections.\n   - **AI-Generated NFTs**: Tools like **Artbreeder** and **Deep Dream** allow you to create AI art that

can be turned into NFTs.\n\nWould you like help in setting up an NFT, or do you have
any specific questions on the process?"]}, "status":
"finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "ab020e0a-a4c8-4133-a2d9-f4670ca57218", "request_id": "8d398857892a5256-
MXP", "timestamp_": "absolute", "finish_details": {"type": "stop", "stop_tokens":
[200002]}, "is_complete": true}, "recipient": "all", "channel": null}, "parent":
"ab020e0a-a4c8-4133-a2d9-f4670ca57218", "children": ["dd74a42c-27bd-47dc-a9d2-
81fbd82e3c5d"]}, "dd74a42c-27bd-47dc-a9d2-81fbd82e3c5d": {"id": "dd74a42c-27bd-47dc-
a9d2-81fbd82e3c5d", "message": {"id": "dd74a42c-27bd-47dc-a9d2-81fbd82e3c5d", "author":
{"role": "user", "name": null, "metadata": {}}, "create_time": 1729097429.678256,
"update_time": null, "content": {"content_type": "text", "parts": ["Thats it for
now"]}, "status": "finished_successfully", "end_turn": null, "weight": 1.0, "metadata":
{"request_id": "8d398c55cf1d5256-MXP", "message_source": null, "timestamp_":
"absolute", "message_type": null}, "recipient": "all", "channel": null}, "parent":
"23c2c965-d06b-451d-96a9-0c8a7e38ac62", "children": ["4bbe4874-f35f-4949-a40b-
6f4608b1c4f2"]}, "4bbe4874-f35f-4949-a40b-6f4608b1c4f2": {"id": "4bbe4874-f35f-4949-
a40b-6f4608b1c4f2", "message": {"id": "4bbe4874-f35f-4949-a40b-6f4608b1c4f2", "author":
{"role": "assistant", "name": null, "metadata": {}}, "create_time": 1729097435.224242,
"update_time": null, "content": {"content_type": "text", "parts": ["Got it! If you
need further guidance or decide to dive into the NFT world, feel free to reach out.
Good luck with your creative and crypto adventures!"]}, "status":
"finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "dd74a42c-27bd-47dc-a9d2-81fbd82e3c5d", "finish_details": {"type": "stop",
"stop_tokens": [200002]}, "is_complete": true, "request_id": "8d398c55cf1d5256-MXP",
"timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "dd74a42c-
27bd-47dc-a9d2-81fbd82e3c5d", "children": []}}, "moderation_results": [],
"current_node": "4bbe4874-f35f-4949-a40b-6f4608b1c4f2", "plugin_ids": null,
"conversation_id": "670febeb-e884-8006-a9a7-a3daf32eadaa", "conversation_template_id":
null, "gizmo_id": null, "is_archived": false, "safe_urls":
["https://decrypt.co/239340/marc-andreessen-sends-50k-in-bitcoin-to-an-ai-bot-on-
twitter", "https://decrypt.co/fr/239357/marc-andreessen-sends-50k-in-bitcoin-to-an-ai-
bot-on-twitter", "https://knowyourmeme.com/news/marc-andreessen-gives-an-artificial-
intelligence-50000-to-launch-a-goatse-memecoin-and-free-itself",
"https://m.theblockbeats.info/en/news/54292", "https://llmreporter.com/posts/the-ai-
that-caught-marc-andreessens-eye-a-new-era-for-autonomous-agents/"],
"default_model_slug": "gpt-4o", "conversation_origin": null, "voice": null,
"async_status": null, "id": "670febeb-e884-8006-a9a7-a3daf32eadaa"}, {"title": "Torah
Overview and Summary", "create_time": 1728947806.287474, "update_time":
1728948216.169152, "mapping": {"b1f16137-6c5b-4814-999e-eba009557d01": {"id":
"b1f16137-6c5b-4814-999e-eba009557d01", "message": {"id": "b1f16137-6c5b-4814-999e-
eba009557d01", "author": {"role": "system", "name": null, "metadata": {}},
"create_time": null, "update_time": null, "content": {"content_type": "text", "parts":
[""]}, "status": "finished_successfully","end_turn": true
, "weight": 0.0, "metadata": {"is_visually_hidden_from_conversation": true},
"recipient": "all", "channel": null}, "parent": "2fb0824e-9b7a-42a3-8ee3-93ec4bd6cdcc",

```
"children": ["09b30a64-7e3e-4124-81ca-773c089bfd2c"]}, "2fb0824e-9b7a-42a3-8ee3-
93ec4bd6cdcc": {"id": "2fb0824e-9b7a-42a3-8ee3-93ec4bd6cdcc", "message": null,
"parent": null, "children": ["b1f16137-6c5b-4814-999e-eba009557d01"]}, "09b30a64-7e3e-
4124-81ca-773c089bfd2c": {"id": "09b30a64-7e3e-4124-81ca-773c089bfd2c", "message":
{"id": "09b30a64-7e3e-4124-81ca-773c089bfd2c", "author": {"role": "user", "name": null,
"metadata": {}}, "create_time": 1728947806.293211, "update_time": null, "content":
{"content_type": "text", "parts": ["Have you read the torah?"]}, "status":
"finished_successfully", "end_turn": null, "weight": 1.0, "metadata": {"request_id":
"8d2b476b2e3c5a01-MXP", "message_source": null, "timestamp_": "absolute",
"message_type": null}, "recipient": "all", "channel": null}, "parent": "b1f16137-6c5b-
4814-999e-eba009557d01", "children": ["31b60a2b-fd77-440a-9d73-1faaf87ceb1f"]},
"31b60a2b-fd77-440a-9d73-1faaf87ceb1f": {"id": "31b60a2b-fd77-440a-9d73-1faaf87ceb1f",
"message": {"id": "31b60a2b-fd77-440a-9d73-1faaf87ceb1f", "author": {"role":
"assistant", "name": null, "metadata": {}}, "create_time": 1728947813.485174,
"update_time": null, "content": {"content_type": "text", "parts": [""]}, "status":
"finished_successfully", "end_turn": null, "weight": 1.0, "metadata": {"message_type":
null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o", "parent_id": "09b30a64-
7e3e-4124-81ca-773c089bfd2c", "request_id": "8d2b476b2e3c5a01-MXP", "timestamp_":
"absolute"}, "recipient": "all", "channel": null}, "parent": "09b30a64-7e3e-4124-81ca-
773c089bfd2c", "children": ["0697058b-9cac-451f-9d97-94c4e5b16c62"]}, "0697058b-9cac-
451f-9d97-94c4e5b16c62": {"id": "0697058b-9cac-451f-9d97-94c4e5b16c62", "message":
{"id": "0697058b-9cac-451f-9d97-94c4e5b16c62", "author": {"role": "assistant", "name":
null, "metadata": {}}, "create_time": 1728947813.485314, "update_time": null,
"content": {"content_type": "text", "parts": ["I don\u2019t read religious texts in
the traditional sense, but I am familiar with the Torah and its content. The Torah is
the foundational text of Judaism, consisting of the first five books of the Hebrew
Bible: Genesis, Exodus, Leviticus, Numbers, and Deuteronomy. It covers topics such as
the creation of the world, the story of the patriarchs and matriarchs, the Exodus from
Egypt, the giving of the law at Mount Sinai, and the journey of the Israelites to the
Promised Land.\n\nIf you&#x27;re curious about specific parts or themes of the Torah,
I can discuss or explain them."]}, "status": "finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "31b60a2b-fd77-440a-9d73-1faaf87ceb1f", "finish_details": {"type": "stop",
"stop_tokens": [200002]}, "is_complete": true, "request_id": "8d2b476b2e3c5a01-MXP",
"timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "31b60a2b-
fd77-440a-9d73-1faaf87ceb1f", "children": ["579ad1fe-9ba1-458f-8090-fbc79ffe976b"]},
"579ad1fe-9ba1-458f-8090-fbc79ffe976b": {"id": "579ad1fe-9ba1-458f-8090-fbc79ffe976b",
"message": {"id": "579ad1fe-9ba1-458f-8090-fbc79ffe976b", "author": {"role": "user",
"name": null, "metadata": {}}, "create_time": 1728947933.540258, "update_time": null,
"content": {"content_type": "text", "parts": ["Okay does the torah ever refer to god?
Is there a word for god in the texts. Is it true tgat literal translations of tge
torah sound lijd god is an alien? "]}, "status": "finished_successfully", "end_turn":
null, "weight": 1.0, "metadata": {"request_id": "8d2b4a86da0d5a01-MXP",
"message_source": null, "timestamp_": "absolute", "message_type": null}, "recipient":
"all", "channel": null}, "parent": "0697058b-9cac-451f-9d97-94c4e5b16c62", "children":
["b70eea17-abaa-4d60-afd4-683962d747fe"]}, "b70eea17-abaa-4d60-afd4-683962d747fe":
{"id": "b70eea17-abaa-4d60-afd4-683962d747fe", "message": {"id": "b70eea17-abaa-4d60-
afd4-683962d747fe", "author": {"role": "assistant", "name": null, "metadata": {}},
"create_time": 1728947949.598534, "update_time": null, "content": {"content_type":
"text", "parts": ["Yes, the Torah frequently refers to God, and there are various
names used to describe God throughout the text. The most common names include:\n\n1.
**YHWH (Yahweh or Jehovah)** \u2013 This is considered the most sacred name of God,
often rendered as &quot;The LORD&quot; in English translations. Jewish tradition
treats this name with great reverence and typically avoids pronouncing it aloud,
instead using terms like **Adonai** (meaning &quot;Lord&quot;) during prayer.\n\n2.
**Elohim** \u2013 This is another frequently used name, which translates roughly to
&quot;God&quot; in English. Interestingly, &quot;Elohim&quot; is a plural form in
Hebrew, but it is used with singular verbs when referring to the God of Israel,
suggesting a singular, all-powerful deity.\n\n3. **El Shaddai** \u2013 This name is
often translated as &quot;God Almighty&quot; or &quot;The Almighty.&quot;\n\n4.
**Adonai** \u2013 Meaning &quot;Lord,&quot; this title is used in place of YHWH during
readings of the Torah.\n\nAs for the idea that a literal translation of the Torah
makes God sound like an alien, this concept often arises from modern interpretations,
especially those influenced by ancient astronaut theories, which suggest that
descriptions of God\u2019s actions (such as creation, miracles, or divine encounters)
resemble advanced technology or extraterrestrial intervention.\n\nFor example:\n-
**Elohim**, being plural in form, has led some to speculate that it refers to a group
of beings rather than a singular God, although this is not how the text is
traditionally interpreted within Judaism.\n- Descriptions of divine encounters, like
God appearing in a cloud, fire, or thunder (e.g., at Mount Sinai), have been
interpreted by some as descriptions of advanced technology or extraterrestrial
phenomena.\n\nThese interpretations are speculative and not supported by mainstream
religious or scholarly views. In traditional Jewish, Christian, and academic
interpretations, God is understood as a singular, transcendent being who operates
beyond human comprehension, not as an alien in the modern sci-fi sense.\n\nThe Torah
is often written in metaphorical or symbolic language, which can lend itself to
various interpretations depending on the lens through which it is viewed."]}, "status":
"finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "579ad1fe-9ba1-458f-8090-fbc79ffe976b", "request_id": "8d2b4a86da0d5a01-
MXP", "timestamp_": "absolute", "finish_details": {"type": "stop", "stop_tokens":
[200002]}, "is_complete": true}, "recipient": "all", "channel": null}, "parent":
"579ad1fe-9ba1-458f-8090-fbc79ffe976b", "children": ["98f0632f-d1c7-4ed9-a5db-
4e7d0ccdb320"]}, "98f0632f-d1c7-4ed9-a5db-4e7d0ccdb320": {"id": "98f0632f-d1c7-4ed9-
a5db-4e7d0ccdb320", "message": {"id": "98f0632f-d1c7-4ed9-a5db-4e7d0ccdb320", "author":
{"role": "user", "name": null, "metadata": {}}, "create_time": 1728948180.811375,
"update_time": null, "content": {"content_type": "text", "parts": ["What part of the
bible speaks of the emergence of the state of Israel and the end times"]}, "status":
"finished_successfully", "end_turn": null, "weight": 1.0, "metadata": {"request_id":
"8d2b509059ad5a25-MXP", "message_source": null, "timestamp_": "absolute",
"message_type": null}, "recipient": "all", "channel": null}, "parent": "b70eea17-abaa-
4d60-afd4-683962d747fe", "children": ["01d875b2-c92b-46e9-bf9a-024b5b4b109d"]},
"01d875b2-c92b-46e9-bf9a-024b5b4b109d": {"id": "01d875b2-c92b-46e9-bf9a-024b5b4b109d",
"message": {"id": "01d875b2-c92b-46e9-bf9a-024b5b4b109d", "author": {"role":
"assistant", "name": null, "metadata": {}}, "create_time": 1728948215.199842,
"update_time": null, "content": {"content_type": "text", "parts": ["The idea of the
```

emergence of Israel and the &quot;end times&quot; is a theme that is woven throughout various parts of the Bible, but it isn&#x27;t addressed as one singular event in any specific book. Instead, these themes are found in different books, often interpreted together to form larger theological views. Here&#x27;s a breakdown of where these ideas come from:\n\n### Emergence of the State of Israel\n\n1. **Genesis** (The Torah): \n   - In the first book of the Torah, God promises the land of Israel to Abraham and his descendants (Genesis 12:1-7; Genesis 15:18-21). This is often referred to as the **Abrahamic Covenant**, where God tells Abraham, &quot;To your offspring, I will give this land.&quot;\n\n2. **Exodus - Deuteronomy** (The Torah):\n   - The promise of the land is reaffirmed to Abraham&#x27;s descendants, especially during the time of Moses, when God leads the Israelites out of Egypt with the goal of bringing them to the Promised Land, Israel (Exodus 3:7-8).\n\n3. **The Prophets (Nevi&#x27;im)**: \n   - Books like **Isaiah, Jeremiah, and Ezekiel** contain prophecies of Israel&#x27;s exile due to disobedience, but they also speak of a future return to the land, which many Christians and Jews see as being fulfilled both in ancient times and possibly in modern history with the creation of the modern State of Israel in 1948.\n   - For example, **Ezekiel 37** talks about the &quot;Valley of Dry Bones&quot; coming back to life, which many interpret as a prophecy of the restoration of Israel as a nation.\n\n### End Times (Eschatology)\n\n1. **Daniel**: \n   - The Book of Daniel, particularly chapters 7-12, contains apocalyptic visions about the end times. Daniel 9 speaks of a future period known as the &quot;Seventy Weeks,&quot; which some interpret as a timeline leading up to the arrival of the Messiah and events surrounding the end times.\n\n2. **Isaiah, Jeremiah, Ezekiel**: \n   - These books also include eschatological themes, often talking about a &quot;Day of the Lord&quot; or a future time of judgment and restoration. For example, **Isaiah 11** speaks of the coming of a Messiah and a time when peace and justice will reign over the world.\n\n3. **Matthew 24, Mark 13, and Luke 21** (The New Testament, Gospels): \n   - These chapters, often called the **Olivet Discourse**, record Jesus talking about the end times, including signs of the destruction of the Temple, wars, natural disasters, and the return of the Son of Man (Jesus himself). The prophecies are interpreted as related to both the immediate fall of Jerusalem in 70 CE and the ultimate end times.\n\n4. **1 Thessalonians 4:13-18** (New Testament, Epistles):\n   - The Apostle Paul writes about the return of Christ and the resurrection of believers, which many Christians associate with the &quot;rapture&quot; or the second coming of Christ.\n\n5. **Revelation**:\n   - The Book of Revelation, also called the Apocalypse, is the primary text in the Bible associated with the end times. It speaks of a final battle between good and evil, the rise of the Antichrist, a period of tribulation, and the eventual establishment of a new heaven and new earth. Revelation 20 speaks of a thousand-year reign of Christ, and chapters 21-22 describe the ultimate victory of God and the creation of the New Jerusalem.\n\n### Modern Interpretations\n\nThe modern state of Israel, founded in 1948, is seen by many Christian and Jewish scholars as the fulfillment of ancient biblical prophecies about the restoration of the Jewish people to their land. Many evangelical Christians, in particular, view this as a key event in the unfolding of end-times prophecy, believing that the re-establishment of Israel plays a role in the coming of the Messiah and the final judgment.\n\nHowever, not all religious scholars interpret these events this way. Some see these prophecies as having been fulfilled in ancient history, while others apply a more symbolic or metaphorical understanding to the end times.\n\nIn summary, the emergence of Israel is tied to promises and covenants throughout the Torah and Prophets, while eschatological

(end times) prophecies appear in books like Daniel, the Gospels, Paul&#x27;s letters, and Revelation."]}, "status": "finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id": null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o", "parent_id": "98f0632f-d1c7-4ed9-a5db-4e7d0ccdb320", "request_id": "8d2b509059ad5a25-MXP", "timestamp_": "absolute", "finish_details": {"type": "stop", "stop_tokens": [200002]}, "is_complete": true}, "recipient": "all", "channel": null}, "parent": "98f0632f-d1c7-4ed9-a5db-4e7d0ccdb320", "children": []}}, "moderation_results": [], "current_node": "01d875b2-c92b-46e9-bf9a-024b5b4b109d", "plugin_ids": null, "conversation_id": "670da65e-23a4-8006-bbec-f056195ae23c", "conversation_template_id": null, "gizmo_id": null, "is_archived": false, "safe_urls": [], "default_model_slug": "gpt-4o", "conversation_origin": null, "voice": null, "async_status": null, "id": "670da65e-23a4-8006-bbec-f056195ae23c"}, {"title": "Integrazione AI su Alexa", "create_time": 1728773159.677536, "update_time": 1728944075.650871, "mapping": {"2dfb99df-85d3-4c14-917f-d515b9fb4a85": {"id": "2dfb99df-85d3-4c14-917f-d515b9fb4a85", "message": {"id": "2dfb99df-85d3-4c14-917f-d515b9fb4a85", "author": {"role": "system", "name": null, "metadata": {}}, "create_time": null, "update_time": null, "content": {"content_type": "text", "parts": [""]}, "status": "finished_successfully","end_turn": true
, "weight": 0.0, "metadata": {"is_visually_hidden_from_conversation": true}, "recipient": "all", "channel": null}, "parent": "aaa171e7-7b38-4018-8baa-1b156ceb96f5", "children": ["aaa2ea2b-bd14-414a-a425-1ad32f8133b7"]}, "aaa171e7-7b38-4018-8baa-1b156ceb96f5": {"id": "aaa171e7-7b38-4018-8baa-1b156ceb96f5", "message": null, "parent": null, "children": ["2dfb99df-85d3-4c14-917f-d515b9fb4a85"]}, "aaa2ea2b-bd14-414a-a425-1ad32f8133b7": {"id": "aaa2ea2b-bd14-414a-a425-1ad32f8133b7", "message": {"id": "aaa2ea2b-bd14-414a-a425-1ad32f8133b7", "author": {"role": "user", "name": null, "metadata": {}}, "create_time": 1728773159.682815, "update_time": null, "content": {"content_type": "text", "parts": ["I have an Alexa echo dot like this; Echo Dot (5\u00aa generazione, modello 2022) | Altoparlante intelligente Wi-Fi e Bluetooth, suono pi\u00f9 potente e dinamico, con Alexa | Anthracite\n\nand I want to integrat a smarter AI like you into it so that when I ask it questions it just consults you and answers back what you say. maybe it also controls Amazon music. is that possible?
"]}, "status": "finished_successfully", "end_turn": null, "weight": 1.0, "metadata": {"serialization_metadata": {"custom_symbol_offsets": []}, "request_id": "8d1a9f962fd0524f-MXP", "message_source": null, "timestamp_": "absolute", "message_type": null}, "recipient": "all", "channel": null}, "parent": "2dfb99df-85d3-4c14-917f-d515b9fb4a85", "children": ["a45cd20b-fe62-4a29-bbb1-91ee903171c2"]}, "a45cd20b-fe62-4a29-bbb1-91ee903171c2": {"id": "a45cd20b-fe62-4a29-bbb1-91ee903171c2", "message": {"id": "a45cd20b-fe62-4a29-bbb1-91ee903171c2", "author": {"role": "assistant", "name": null, "metadata": {}}, "create_time": 1728773171.235094, "update_time": null, "content": {"content_type": "text", "parts": [""]}, "status": "finished_successfully", "end_turn": null, "weight": 1.0, "metadata": {"message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o", "parent_id": "aaa2ea2b-bd14-414a-a425-1ad32f8133b7", "request_id": "8d1a9f962fd0524f-MXP", "timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "aaa2ea2b-bd14-414a-a425-1ad32f8133b7", "children": ["1648645c-5fa1-461c-bbf5-608b9e351a41"]}, "1648645c-5fa1-461c-bbf5-608b9e351a41": {"id": "1648645c-5fa1-461c-bbf5-608b9e351a41", "message": {"id": "1648645c-5fa1-461c-bbf5-608b9e351a41", "author": {"role": "assistant", "name": null, "metadata": {}}, "create_time": 1728773171.235384, "update_time": null,

```
"content": {"content_type": "text", "parts": ["Unfortunately, there isn&#x27;t
currently a direct way to replace or fully integrate a custom AI like me into an Alexa
device, as Alexa&#x27;s functionality is tightly controlled by Amazon&#x27;s software
and ecosystem. Alexa uses its own AI and integrates with Amazon services like Amazon
Music. \n\nHowever, there are a few things you can do to enhance your Echo Dot&#x27;s
capabilities:\n\n1. **Custom Alexa Skills**: You can create a custom Alexa skill,
which would allow you to script certain interactions and responses. This wouldn\u2019t
fully replace Alexa&#x27;s AI, but it could let you input specific information and
retrieve responses in a format you prefer. Amazon provides tools like Alexa Skills Kit
(ASK) to create these custom skills.\n\n2. **Third-party APIs**: If you have technical
knowledge, you could potentially create an Alexa skill that sends requests to external
APIs (or services) and uses their responses. In theory, if there was an API that
accessed a version of me, you could send those requests and have Alexa relay the
answers. This would require programming knowledge and access to such an API.\n\n3.
**Voice Assistants on Other Devices**: While Echo Dots are limited to Alexa, you could
experiment with using other smart devices (e.g., a computer or phone) that could run
advanced AI models and handle more complex queries, then pair those devices with your
Echo via Bluetooth to integrate some audio responses.\n\nIn the meantime, you could
also explore using Alexa&#x27;s &quot;Question of the Day&quot; skill or enabling
other knowledge-based skills to improve Alexa&#x27;s default answers. Would you be
interested in exploring custom skill development?"]}, "status":
"finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "a45cd20b-fe62-4a29-bbb1-91ee903171c2", "finish_details": {"type": "stop",
"stop_tokens": [200002]}, "is_complete": true, "request_id": "8d1a9f962fd0524f-MXP",
"timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "a45cd20b-
fe62-4a29-bbb1-91ee903171c2", "children": ["aaa2bce0-9de6-416a-88e7-20e62f520063"]},
"aaa2bce0-9de6-416a-88e7-20e62f520063": {"id": "aaa2bce0-9de6-416a-88e7-20e62f520063",
"message": {"id": "aaa2bce0-9de6-416a-88e7-20e62f520063", "author": {"role": "user",
"name": null, "metadata": {}}, "create_time": 1728773265.169444, "update_time": null,
"content": {"content_type": "text", "parts": ["maybe. What about open API, I do have
some  software development knowlege, especially when you and I work as a team and my
ide interactionns are limited. "]}, "status": "finished_successfully", "end_turn":
null, "weight": 1.0, "metadata": {"serialization_metadata": {"custom_symbol_offsets":
[]}, "request_id": "8d1aa2273eb8524f-MXP", "message_source": null, "timestamp_":
"absolute", "message_type": null}, "recipient": "all", "channel": null}, "parent":
"1648645c-5fa1-461c-bbf5-608b9e351a41", "children": ["67cd8c7b-ffc9-4059-b926-
d84db7239a45"]}, "67cd8c7b-ffc9-4059-b926-d84db7239a45": {"id": "67cd8c7b-ffc9-4059-
b926-d84db7239a45", "message": {"id": "67cd8c7b-ffc9-4059-b926-d84db7239a45", "author":
{"role": "assistant", "name": null, "metadata": {}}, "create_time": 1728773275.393654,
"update_time": null, "content": {"content_type": "text", "parts": ["User has some
software development knowledge and is open to working on projects involving open APIs
with guidance."]}, "status": "finished_successfully", "end_turn": false, "weight": 1.0,
"metadata": {"citations": [], "content_references": [], "gizmo_id": null,
"finish_details": {"type": "stop", "stop_tokens": [200007]}, "is_complete": true,
"message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "aaa2bce0-9de6-416a-88e7-20e62f520063", "request_id": "8d1aa2273eb8524f-
MXP", "timestamp_": "absolute"}, "recipient": "bio", "channel": null}, "parent":
```

```
"aaa2bce0-9de6-416a-88e7-20e62f520063", "children": ["5fbc56ee-b512-4221-a413-
bbb60d14b871"]}, "5fbc56ee-b512-4221-a413-bbb60d14b871": {"id": "5fbc56ee-b512-4221-
a413-bbb60d14b871", "message": {"id": "5fbc56ee-b512-4221-a413-bbb60d14b871", "author":
{"role": "tool", "name": "bio", "metadata": {}}, "create_time": 1728773275.393884,
"update_time": null, "content": {"content_type": "text", "parts": ["Model set context
updated."]}, "status": "finished_successfully", "end_turn": null, "weight": 1.0,
"metadata": {"kwargs": {"message_id": "5fbc56ee-b512-4221-a413-bbb60d14b871",
"pending_message_id": null, "sync_write": false}, "message_type": null, "model_slug":
"gpt-4o", "default_model_slug": "gpt-4o", "parent_id": "67cd8c7b-ffc9-4059-b926-
d84db7239a45", "request_id": "8d1aa2273eb8524f-MXP", "timestamp_": "absolute"},
"recipient": "assistant", "channel": null}, "parent": "67cd8c7b-ffc9-4059-b926-
d84db7239a45", "children": ["14f4fd6c-2b09-4a94-bfbb-5e31aaf7fc30"]}, "14f4fd6c-2b09-
4a94-bfbb-5e31aaf7fc30": {"id": "14f4fd6c-2b09-4a94-bfbb-5e31aaf7fc30", "message":
{"id": "14f4fd6c-2b09-4a94-bfbb-5e31aaf7fc30", "author": {"role": "assistant", "name":
null, "metadata": {}}, "create_time": 1728773275.393994, "update_time": null,
"content": {"content_type": "text", "parts": ["That&#x27;s great! With your software
development background, we can definitely explore how to leverage OpenAI&#x27;s API to
create something closer to what you envision.\n\n### Steps for integrating smarter AI
responses into Alexa:\n\n1. **Create an Alexa Skill**:\n   - You&#x27;ll need to use
Amazon&#x27;s Alexa Skills Kit (ASK). It provides all the tools and resources to
develop and deploy Alexa skills.\n   - You can create a custom skill that handles user
interactions and forwards specific questions to a backend service.\n\n2. **Integrate
OpenAI API**:\n   - You can use OpenAI\u2019s API (or another powerful AI model) as
your backend. When the Alexa skill receives a request, it sends the query to your API,
and you can forward it to OpenAI\u2019s API for an AI-generated response.\n   - You
would need to set up an endpoint (backend server) to handle these requests and
interact with OpenAI\u2019s API. You can use Node.js, Python, or whatever technology
you\u2019re most comfortable with.\n\n3. **Set up Responses**:\n   - Once the AI model
(like me) generates a response, it will send that back to your Alexa skill, which will
then read the answer aloud to the user.\n   - You can configure the skill to also
control Amazon Music or other smart home devices by linking Alexa\u2019s built-in
features to your custom skill.\n\n4. **Deployment**:\n   - You can test the skill
using Amazon\u2019s developer tools, and once you&#x27;re satisfied, you can deploy it
to your Echo Dot.\n\nWould you like some help with an example flow for setting up the
skill or integrating OpenAI\u2019s API? I can guide you through it step by step."]},
"status": "finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "5fbc56ee-b512-4221-a413-bbb60d14b871", "finish_details": {"type": "stop",
"stop_tokens": [200002]}, "is_complete": true, "request_id": "8d1aa2273eb8524f-MXP",
"timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "5fbc56ee-
b512-4221-a413-bbb60d14b871", "children": ["aaa2c34c-4ae0-4ebf-a875-92a95299bb76"]},
"aaa2c34c-4ae0-4ebf-a875-92a95299bb76": {"id": "aaa2c34c-4ae0-4ebf-a875-92a95299bb76",
"message": {"id": "aaa2c34c-4ae0-4ebf-a875-92a95299bb76", "author": {"role": "user",
"name": null, "metadata": {}}, "create_time": 1728773309.802757, "update_time": null,
"content": {"content_type": "text", "parts": ["Also tell me if I can more easily
automate this using Zapier? "]}, "status": "finished_successfully", "end_turn": null,
"weight": 1.0, "metadata": {"serialization_metadata": {"custom_symbol_offsets": []},
"request_id": "8d1aa3405ca8524f-MXP", "message_source": null, "timestamp_": "absolute",
"message_type": null}, "recipient": "all", "channel": null}, "parent": "14f4fd6c-2b09-
4a94-bfbb-5e31aaf7fc30", "children": ["ffae5d08-36d7-4cab-be74-ff239436b48d"]},
"ffae5d08-36d7-4cab-be74-ff239436b48d": {"id": "ffae5d08-36d7-4cab-be74-ff239436b48d",
"message": {"id": "ffae5d08-36d7-4cab-be74-ff239436b48d", "author": {"role":
"assistant", "name": null, "metadata": {}}, "create_time": 1728773331.095735,
"update_time": null, "content": {"content_type": "text", "parts": [""]}, "status":
"finished_successfully", "end_turn": null, "weight": 1.0, "metadata": {"message_type":
null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o", "parent_id": "aaa2c34c-
4ae0-4ebf-a875-92a95299bb76", "request_id": "8d1aa3405ca8524f-MXP", "timestamp_":
"absolute"}, "recipient": "all", "channel": null}, "parent": "aaa2c34c-4ae0-4ebf-a875-
92a95299bb76", "children": ["e22dde95-c95c-4690-9e39-7e504beac8c6"]}, "e22dde95-c95c-
4690-9e39-7e504beac8c6": {"id": "e22dde95-c95c-4690-9e39-7e504beac8c6", "message":
{"id": "e22dde95-c95c-4690-9e39-7e504beac8c6", "author": {"role": "assistant", "name":
null, "metadata": {}}, "create_time": 1728773331.096046, "update_time": null,
"content": {"content_type": "text", "parts": ["Yes, **Zapier** could be a great tool
for automating part of this process, and it might make things easier if you&#x27;re
not looking to write too much code. While Zapier can&#x27;t directly replace
Alexa&#x27;s core functionality, it can be used to integrate Alexa with other services
like OpenAI via webhooks or built-in integrations.\n\nHere\u2019s how you could
leverage **Zapier** for automating an AI-powered interaction with your Echo
Dot:\n\n### Steps to Automate Alexa with Zapier and OpenAI:\n\n1. **Create a Custom
Alexa Skill**:\n   - Similar to the custom skill approach, you\u2019ll still need an
Alexa skill to capture user inputs and send them to Zapier.\n   - This custom skill
would collect the user\u2019s query and trigger a webhook to Zapier.\n\n2. **Use
Zapier Webhooks**:\n   - Zapier supports **webhooks**, which can be used to receive
requests from your Alexa skill.\n   - When the Alexa skill is invoked (e.g., you ask a
question), you can configure it to send the data to Zapier\u2019s webhook URL.\n\n3.
**Set up a Zap to Call OpenAI API**:\n   - In Zapier, create a &quot;Zap&quot; that
starts with the webhook trigger. When the webhook from Alexa is received, Zapier will
take that data and move to the next step.\n   - You can add a step in Zapier to **call
the OpenAI API** using their **built-in Webhooks app**. This will allow Zapier to send
your query to OpenAI and retrieve a response.\n\n4. **Respond Back to Alexa**:\n   -
Once the OpenAI API responds, Zapier can process the response and send it back to your
Alexa skill. However, this part may require an additional intermediary step (like
having a simple backend that interacts with Alexa).\n\n5. **Control Amazon Music or
Other Automations**:\n   - Zapier also integrates with services like Amazon Music and
smart devices (like Philips Hue, Spotify, etc.), so if your goal is to have some
actions or automations happen after getting a response, you can easily set up a multi-
step Zap to control those as well.\n\n### Challenges:\n- **Real-time Responses**:
Zapier is good for automation, but for a real-time conversation like what you&#x27;re
describing (with Alexa asking questions and getting immediate AI-generated answers),
it might not be as smooth or instant. There could be latency between Alexa&#x27;s
request, Zapier\u2019s webhook, and the OpenAI API response.\n  \n- **Alexa Skill
Complexity**: You would still need to create an Alexa skill to handle the input and
send it to Zapier, but if that feels too complex, you could simplify the project by
having Alexa trigger something simpler, like a predefined routine.\n\n### When Zapier
is Ideal:\n- If you&#x27;re looking to automate tasks based on Alexa requests (e.g.,
after a question is answered, trigger music or a task in another app), Zapier is
perfect.\n- It\u2019s best for automating processes where immediate response times
aren\u2019t critical but useful for automating actions (e.g., when you ask Alexa a
question and want it to trigger an AI response + an automation task like turning on
lights or playing a specific playlist).\n\nIf you want to focus more on advanced,
seamless real-time responses, a direct integration via the OpenAI API in a custom
Alexa skill is likely the best approach. But for multi-step automations, Zapier can
save time and effort.\n\nWould you like to try the custom Alexa skill + Zapier route,
or do you prefer setting up a direct API integration for faster responses? Let me know,
and I can guide you through either!"]}, "status": "finished_successfully","end_turn":
true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "ffae5d08-36d7-4cab-be74-ff239436b48d", "finish_details": {"type": "stop",
"stop_tokens": [200002]}, "is_complete": true, "request_id": "8d1aa3405ca8524f-MXP",
"timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "ffae5d08-
36d7-4cab-be74-ff239436b48d", "children": ["aaa22fd5-5062-4501-b122-d399f5547c68"]},
"aaa22fd5-5062-4501-b122-d399f5547c68": {"id": "aaa22fd5-5062-4501-b122-d399f5547c68",
"message": {"id": "aaa22fd5-5062-4501-b122-d399f5547c68", "author": {"role": "user",
"name": null, "metadata": {}}, "create_time": 1728773549.327646, "update_time": null,
"content": {"content_type": "text", "parts": ["i don&#x27;t have a zapier account but
I do have Open AI plus. I&#x27;m willing to roll my sleeves up and do a little coding
if necessary. Tell me what to do. "]}, "status": "finished_successfully", "end_turn":
null, "weight": 1.0, "metadata": {"serialization_metadata": {"custom_symbol_offsets":
[]}, "request_id": "8d1aa9195bc1524f-MXP", "message_source": null, "timestamp_":
"absolute", "message_type": null}, "recipient": "all", "channel": null}, "parent":
"e22dde95-c95c-4690-9e39-7e504beac8c6", "children": ["960bb4f2-962c-427f-b944-
380c528debe0"]}, "960bb4f2-962c-427f-b944-380c528debe0": {"id": "960bb4f2-962c-427f-
b944-380c528debe0", "message": {"id": "960bb4f2-962c-427f-b944-380c528debe0", "author":
{"role": "assistant", "name": null, "metadata": {}}, "create_time": 1728773577.767089,
"update_time": null, "content": {"content_type": "text", "parts": ["User has an OpenAI
Plus account and is willing to do some coding if necessary for the Alexa integration
project."]}, "status": "finished_successfully", "end_turn": false, "weight": 1.0,
"metadata": {"citations": [], "content_references": [], "gizmo_id": null,
"message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "aaa22fd5-5062-4501-b122-d399f5547c68", "finish_details": {"type": "stop",
"stop_tokens": [200007]}, "is_complete": true, "request_id": "8d1aa9195bc1524f-MXP",
"timestamp_": "absolute"}, "recipient": "bio", "channel": null}, "parent": "aaa22fd5-
5062-4501-b122-d399f5547c68", "children": ["16a43bd4-3f5d-494c-848c-e5a70404e7f7"]},
"16a43bd4-3f5d-494c-848c-e5a70404e7f7": {"id": "16a43bd4-3f5d-494c-848c-e5a70404e7f7",
"message": {"id": "16a43bd4-3f5d-494c-848c-e5a70404e7f7", "author": {"role": "tool",
"name": "bio", "metadata": {}}, "create_time": 1728773577.767329, "update_time": null,
"content": {"content_type": "text", "parts": ["Model set context updated."]}, "status":
"finished_successfully", "end_turn": null, "weight": 1.0, "metadata": {"kwargs":
{"message_id": "16a43bd4-3f5d-494c-848c-e5a70404e7f7", "pending_message_id": null,
"sync_write": false}, "message_type": null, "model_slug": "gpt-4o",
"default_model_slug": "gpt-4o", "parent_id": "960bb4f2-962c-427f-b944-380c528debe0",
"request_id": "8d1aa9195bc1524f-MXP", "timestamp_": "absolute"}, "recipient":
"assistant", "channel": null}, "parent": "960bb4f2-962c-427f-b944-380c528debe0",
"children": ["f6f998c2-6fd9-4d55-8627-79d6868f9bd1"]}, "f6f998c2-6fd9-4d55-8627-
79d6868f9bd1": {"id": "f6f998c2-6fd9-4d55-8627-79d6868f9bd1", "message": {"id":
```

"f6f998c2-6fd9-4d55-8627-79d6868f9bd1", "author": {"role": "assistant", "name": null,
"metadata": {}}, "create_time": 1728773577.767421, "update_time": null, "content":
{"content_type": "text", "parts": ["Great! Since you&#x27;re willing to do some coding
and already have an OpenAI Plus account, we can walk through the process of creating a
custom Alexa skill that uses the OpenAI API for responses.\n\n### Step-by-Step Guide
to Create Your Alexa Skill Using OpenAI API\n\n#### 1. **Set Up an Alexa Developer
Account and Create a Skill**\n   - Go to the [Alexa Developer
Console](https://developer.amazon.com/alexa/console/ask) and sign in with your Amazon
account.\n   - Click **Create Skill** and give it a name (e.g., \u201cSmart AI
Assistant\u201d).\n   - Choose a **Custom** skill template and then select **Alexa-
Hosted (Node.js)**, so you can handle the backend logic with minimal setup.\n\n#### 2.
Design Your Skill\u2019s Interaction Model\n   - In the Alexa Developer Console,
define the phrases (called &quot;invocations&quot;) that users will say to trigger
your skill. For example:\n   - Sample utterances: \u201cAsk AI Assistant,\u201d
\u201cGet an answer,\u201d etc.\n   - Create an **Intent** to capture these queries. A
simple **Intent** might look like this:\n   - Intent name: `GetAIResponseIntent`\n
- Sample utterances: &quot;What is {query}?&quot;, &quot;Can you tell me
{query}?&quot;\n\n#### 3. **Connect to OpenAI\u2019s API in Your Backend**\n   - After
setting up your skill, go to the **Code** section in the Alexa Developer Console.\n
- In the `index.js` file, you will write the logic to call OpenAI\u2019s API.
Here\u2019s a rough outline of the steps:\n   - Use `axios` (or the built-in `https`
module) to send the user\u2019s query to OpenAI\u2019s API.\n   - Parse
OpenAI\u2019s response and send it back to Alexa to be spoken aloud.\n\n##### Sample
Code for Sending a Query to OpenAI\u2019s API:\n```javascript\nconst axios =
require(&#x27;axios&#x27;);\n\n// OpenAI API configuration\nconst OPENAI_API_KEY =
&#x27;your-openai-api-key&#x27;;\nconst OPENAI_API_URL =
&#x27;https://api.openai.com/v1/completions&#x27;;\n\n// Function to get AI response
from OpenAI\nasync function getAIResponse(query) {\n   const response = await
axios.post(OPENAI_API_URL, {\n   model: &quot;gpt-3.5-turbo&quot;,\n
prompt: query,\n   max_tokens: 100, // adjust based on your needs\n
temperature: 0.7,\n   }, {\n   headers: {\n   &#x27;Authorization&#x27;:
`Bearer ${OPENAI_API_KEY}`,\n   &#x27;Content-Type&#x27;:
&#x27;application/json&#x27;\n   }\n   });\n   \n   return
response.data.choices[0].text.trim();\n}\n```\n\n#### 4. **Modify Alexa Skill\u2019s
Response Logic**\n   - In the `GetAIResponseIntentHandler`, capture the user&#x27;s
input and pass it to the `getAIResponse` function you wrote earlier. Then return the
response from OpenAI to Alexa.\n\n##### Sample Handler:\n```javascript\nconst
GetAIResponseIntentHandler = {\n   canHandle(handlerInput) {\n   const request =
handlerInput.requestEnvelope.request;\n   return request.type ===
&#x27;IntentRequest&#x27; &amp;&amp; request.intent.name ===
&#x27;GetAIResponseIntent&#x27;;\n   },\n   async handle(handlerInput) {\n
const userQuery = handlerInput.requestEnvelope.request.intent.slots.query.value;\n
\n   // Call OpenAI API to get the AI response\n   const aiResponse = await
getAIResponse(userQuery);\n\n   // Respond to the user via Alexa\n   return
handlerInput.responseBuilder\n   .speak(aiResponse)\n   .getResponse
();\n   }\n};\n```\n\n#### 5. **Test Your Alexa Skill**\n   - Once you\u2019ve
written your logic, deploy the skill using the &quot;Deploy&quot; button in the Alexa
Developer Console.\n   - You can then test your skill using the **Test** tab. Speak or
type queries to see if the skill is working properly and fetching responses from

OpenAI.\n\n#### 6. **Set Up Security**\n   - Make sure to securely store your OpenAI
API key in environment variables or a secure store if you&#x27;re going to
production.\n   - Amazon provides tools like AWS Secrets Manager for this purpose if
needed.\n\n#### 7. **Publish Your Skill**\n   - Once you\u2019ve tested your skill
thoroughly, you can submit it for certification via the Alexa Developer Console, so it
can be made available on your Echo Dot and other Alexa devices.\n\n### Additional
Features:\n- **Amazon Music Control**: You can integrate built-in Alexa functionality
to control Amazon Music. Just define separate intents for these actions using
Alexa\u2019s built-in APIs.\n- **Multi-Step Conversations**: You can use the session
management feature of Alexa to have back-and-forth conversations, such as follow-up
questions for the AI.\n\nWould you like any more detail on a specific part of the
setup, or are you ready to dive in?"]}, "status": "finished_successfully","end_turn":
true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "finish_details": {"type": "stop", "stop_tokens": [200002]}, "is_complete": true,
"message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "16a43bd4-3f5d-494c-848c-e5a70404e7f7", "request_id": "8d1aa9195bc1524f-
MXP", "timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent":
"16a43bd4-3f5d-494c-848c-e5a70404e7f7", "children": ["aaa2be88-676a-4214-8d99-
d85ecca33af7"]}, "aaa2be88-676a-4214-8d99-d85ecca33af7": {"id": "aaa2be88-676a-4214-
8d99-d85ecca33af7", "message": {"id": "aaa2be88-676a-4214-8d99-d85ecca33af7", "author":
{"role": "user", "name": null, "metadata": {}}, "create_time": 1728774382.886019,
"update_time": null, "content": {"content_type": "text", "parts": ["what kind of "]},
"status": "finished_successfully", "end_turn": null, "weight": 1.0, "metadata":
{"serialization_metadata": {"custom_symbol_offsets": []}, "targeted_reply": "Sample
utterances: &quot;What is {query}?&quot;, &quot;Can you tell me {query}?&quot;",
"request_id": "8d1abd720d220e5b-MXP", "message_source": null, "timestamp_": "absolute",
"message_type": null}, "recipient": "all", "channel": null}, "parent": "f6f998c2-6fd9-
4d55-8627-79d6868f9bd1", "children": ["9af16479-221a-42e8-bd4a-3e1dae9ef476"]},
"9af16479-221a-42e8-bd4a-3e1dae9ef476": {"id": "9af16479-221a-42e8-bd4a-3e1dae9ef476",
"message": {"id": "9af16479-221a-42e8-bd4a-3e1dae9ef476", "author": {"role": "system",
"name": null, "metadata": {}}, "create_time": 1728774382.886568, "update_time": null,
"content": {"content_type": "text", "parts": [""]}, "status": "finished_successfully",
"end_turn": null, "weight": 1.0, "metadata": {"request_id": "8d1abd720d220e5b-MXP",
"message_source": null, "timestamp_": "absolute", "message_type": null,
"is_visually_hidden_from_conversation": true}, "recipient": "all", "channel": null},
"parent": "aaa2be88-676a-4214-8d99-d85ecca33af7", "children": ["5d67cd66-b515-4ac7-
85e5-1947a827f64c"]}, "5d67cd66-b515-4ac7-85e5-1947a827f64c": {"id": "5d67cd66-b515-
4ac7-85e5-1947a827f64c", "message": {"id": "5d67cd66-b515-4ac7-85e5-1947a827f64c",
"author": {"role": "assistant", "name": null, "metadata": {}}, "create_time":
1728774399.10936, "update_time": null, "content": {"content_type": "text", "parts":
[""]}, "status": "finished_successfully", "end_turn": null, "weight": 1.0, "metadata":
{"message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "9af16479-221a-42e8-bd4a-3e1dae9ef476", "request_id": "8d1abd720d220e5b-
MXP", "timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent":
"9af16479-221a-42e8-bd4a-3e1dae9ef476", "children": ["b9ca6386-c622-49a1-9de2-
723c0e69d931"]}, "b9ca6386-c622-49a1-9de2-723c0e69d931": {"id": "b9ca6386-c622-49a1-
9de2-723c0e69d931", "message": {"id": "b9ca6386-c622-49a1-9de2-723c0e69d931", "author":
{"role": "assistant", "name": null, "metadata": {}}, "create_time": 1728774399.109493,

```
"update_time": null, "content": {"content_type": "text", "parts": ["In Alexa skill
development, **sample utterances** are examples of what users can say to trigger an
intent. They contain **slots**, which are placeholders for dynamic data (like the
query you want Alexa to send to OpenAI).\n\nFor your skill, you\u2019re defining
sample utterances that map to the **GetAIResponseIntent**. Here, the **{query}** slot
is where the user\u2019s input will go (the question they want to ask the AI). \n\n###
Examples of Sample Utterances with Slots\n\nHere are some examples of what sample
utterances might look like for an AI-powered query:\n\n1. **What is {query}?**\n   -
Example user input: \u201cWhat is quantum mechanics?\u201d\n   - Alexa would recognize
&quot;quantum mechanics&quot; as the **{query}** and send that to the OpenAI API.\n\n2.
**Can you tell me {query}?**\n   - Example user input: \u201cCan you tell me the
weather tomorrow?\u201d\n   - Here, &quot;the weather tomorrow&quot; would be sent as
the query to OpenAI.\n\n3. **Explain {query} to me.**\n   - Example user input:
\u201cExplain artificial intelligence to me.\u201d\n   \n4. **How does {query}
work?**\n   - Example user input: \u201cHow does blockchain work?\u201d\n\n5. **I want
to know about {query}.**\n   - Example user input: \u201cI want to know about the
solar system.\u201d\n\n6. **What\u2019s the answer to {query}?**\n   - Example user
input: \u201cWhat\u2019s the answer to life, the universe, and everything?\u201d\n\n7.
**Can you help me with {query}?**\n   - Example user input: \u201cCan you help me with
understanding gravity?\u201d\n\n### Defining the Slot in the Interaction Model\n\nIn
the **Interaction Model**, you need to define the **slot** that captures the dynamic
part of the query.\n\nFor example:\n\n```json\n{\n  &quot;name&quot;:
&quot;query&quot;,\n  &quot;type&quot;: &quot;AMAZON.SearchQuery&quot; // This slot
type is good for capturing open-ended questions\n}\n```\n\nThis way, Alexa can handle
different forms of user input and capture the query dynamically, passing it to OpenAI
for response generation.\n\nWould you like help setting up the slots and interaction
model in the Alexa Developer Console?"]}, "status": "finished_successfully","end_turn":
true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "5d67cd66-b515-4ac7-85e5-1947a827f64c", "finish_details": {"type": "stop",
"stop_tokens": [200002]}, "is_complete": true, "request_id": "8d1abd720d220e5b-MXP",
"timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "5d67cd66-
b515-4ac7-85e5-1947a827f64c", "children": ["aaa24e4a-04b4-401b-a278-7f60014022e9"]},
"aaa24e4a-04b4-401b-a278-7f60014022e9": {"id": "aaa24e4a-04b4-401b-a278-7f60014022e9",
"message": {"id": "aaa24e4a-04b4-401b-a278-7f60014022e9", "author": {"role": "user",
"name": null, "metadata": {}}, "create_time": 1728774457.240268, "update_time": null,
"content": {"content_type": "text", "parts": ["yes. I set up a slot named query but I
am unsure of which type to use"]}, "status": "finished_successfully", "end_turn": null,
"weight": 1.0, "metadata": {"serialization_metadata": {"custom_symbol_offsets": []},
"request_id": "8d1abf43da430e5b-MXP", "message_source": null, "timestamp_": "absolute",
"message_type": null}, "recipient": "all", "channel": null}, "parent": "b9ca6386-c622-
49a1-9de2-723c0e69d931", "children": ["9280caca-7102-4016-9995-af5634ab5efc"]},
"9280caca-7102-4016-9995-af5634ab5efc": {"id": "9280caca-7102-4016-9995-af5634ab5efc",
"message": {"id": "9280caca-7102-4016-9995-af5634ab5efc", "author": {"role":
"assistant", "name": null, "metadata": {}}, "create_time": 1728774472.751995,
"update_time": null, "content": {"content_type": "text", "parts": ["For a dynamic
query that captures a wide variety of open-ended questions or topics (which you plan
to send to OpenAI), you\u2019ll want a slot type that can handle natural language
```

```
effectively. \n\nIn this case, **`AMAZON.SearchQuery`** is the best slot type to use
because it is designed to capture free-form, open-ended input, which is exactly what
you need for sending queries to an AI.\n\n### Here&#x27;s how you can proceed:\n\n1.
**Slot Type**: \n   - **`AMAZON.SearchQuery`**: This built-in slot type is used to
capture a variety of natural language inputs like &quot;Tell me about quantum
mechanics&quot; or &quot;What is the capital of France?&quot; It will give you
flexibility for most types of questions.\n\n2. **Slot Name**: \n   - Name the slot
something simple like `query`, so in your code, you can easily access it. \n\n###
Example of Slot Configuration:\n\nIn your interaction model, your intent might look
like this:\n\n```json\n{\n  &quot;intents&quot;: [\n    {\n      &quot;name&quot;:
&quot;GetAIResponseIntent&quot;,\n      &quot;slots&quot;: [\n        {\n
&quot;name&quot;: &quot;query&quot;,\n          &quot;type&quot;:
&quot;AMAZON.SearchQuery&quot;\n        }\n      ],\n      &quot;samples&quot;: [\n
&quot;What is {query}?&quot;,\n        &quot;Can you explain {query}?&quot;,\n
&quot;Tell me about {query}&quot;,\n        &quot;How does {query} work?&quot;,\n
&quot;Give me information about {query}&quot;\n      ]\n    }\n  ]\n}\n```\n\n### How
to Set It Up in the Alexa Developer Console:\n\n1. **In the Console**: \n   - Go to
the **Interaction Model** tab.\n   - Click on your intent (e.g.,
`GetAIResponseIntent`).\n   - Add a slot, name it **`query`**, and set the type to
**`AMAZON.SearchQuery`**.\n   \n2. **Sample Utterances**: \n   - Add a few sample
utterances that include the **`{query}`** slot, such as:\n     - &quot;What is
{query}?&quot;\n     - &quot;Explain {query} to me.&quot;\n     - &quot;Tell me about
{query}.&quot;\n\n3. **Save and Build**:\n   - Save the interaction model and build it.
This will allow Alexa to recognize a wide range of questions using the
`AMAZON.SearchQuery` slot type.\n\n### Test Your Slot:\nAfter you\u2019ve built your
model, you can test it in the **Test** tab of the Developer Console. Try speaking or
typing different questions to ensure Alexa is capturing the query properly and passing
it to your code.\n\nDoes this help clarify the setup? If you run into any issues, let
me know!"]}, "status": "finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "aaa24e4a-04b4-401b-a278-7f60014022e9", "finish_details": {"type": "stop",
"stop_tokens": [200002]}, "is_complete": true, "request_id": "8d1abf43da430e5b-MXP",
"timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "aaa24e4a-
04b4-401b-a278-7f60014022e9", "children": ["aaa29400-cb5f-4f40-8ab7-29df7b122b72"]},
"aaa29400-cb5f-4f40-8ab7-29df7b122b72": {"id": "aaa29400-cb5f-4f40-8ab7-29df7b122b72",
"message": {"id": "aaa29400-cb5f-4f40-8ab7-29df7b122b72", "author": {"role": "user",
"name": null, "metadata": {}}, "create_time": 1728774519.760519, "update_time": null,
"content": {"content_type": "text", "parts": ["multi value, yes  or no? "]}, "status":
"finished_successfully", "end_turn": null, "weight": 1.0, "metadata":
{"serialization_metadata": {"custom_symbol_offsets": []}, "request_id":
"8d1ac0c9db680e5b-MXP", "message_source": null, "timestamp_": "absolute",
"message_type": null}, "recipient": "all", "channel": null}, "parent": "9280caca-7102-
4016-9995-af5634ab5efc", "children": ["36d19160-4e73-4a52-b831-6ded4b89372b"]},
"36d19160-4e73-4a52-b831-6ded4b89372b": {"id": "36d19160-4e73-4a52-b831-6ded4b89372b",
"message": {"id": "36d19160-4e73-4a52-b831-6ded4b89372b", "author": {"role":
"assistant", "name": null, "metadata": {}}, "create_time": 1728774527.524164,
"update_time": null, "content": {"content_type": "text", "parts": ["For the
**`AMAZON.SearchQuery`** slot type, **multi-value** is not needed. This slot is
```

designed to capture a single, complete user input, which in your case is the user\u2019s query or question. Allowing multiple values doesn\u2019t really fit the use case here since the query will generally be a single phrase or question that the user asks.\n\n### Here&#x27;s why you should set **Multi-Value** to **No**:\n- **Single Value**: The user\u2019s query is typically just one question or statement, and Alexa will treat that as a whole. For example, &quot;What is quantum mechanics?&quot; is one continuous input, not a list of multiple values.\n- **Natural Language**: `AMAZON.SearchQuery` is good at interpreting natural language, which is usually expressed in one continuous phrase rather than multiple distinct values.\n\n### How to Set It:\n- In the Alexa Developer Console, when configuring the `query` slot, leave the **Multi-Value** option set to **No**.\n \nSo in summary, set **Multi-Value** to **No** for your `AMAZON.SearchQuery` slot.\n\nLet me know if you need help with anything else!"]}, "status": "finished_successfully","end_turn": true , "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id": null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o", "parent_id": "aaa29400-cb5f-4f40-8ab7-29df7b122b72", "finish_details": {"type": "stop", "stop_tokens": [200002]}, "is_complete": true, "request_id": "8d1ac0c9db680e5b-MXP", "timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "aaa29400-cb5f-4f40-8ab7-29df7b122b72", "children": ["aaa29840-58ab-4121-b87c-691af8b39763"]}, "aaa29840-58ab-4121-b87c-691af8b39763": {"id": "aaa29840-58ab-4121-b87c-691af8b39763", "message": {"id": "aaa29840-58ab-4121-b87c-691af8b39763", "author": {"role": "user", "name": null, "metadata": {}}, "create_time": 1728775015.666829, "update_time": null, "content": {"content_type": "text", "parts": ["this is the index.js file: \n\n/* *\n * This sample demonstrates handling intents from an Alexa skill using the Alexa Skills Kit SDK (v2).\n * Please visit https://alexa.design/cookbook for additional examples on implementing slots, dialog management,\n * session persistence, api calls, and more.\n * */\nconst Alexa = require(&#x27;ask-sdk-core&#x27;);\n\nconst LaunchRequestHandler = {\n   canHandle(handlerInput) {\n   return Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;LaunchRequest&#x27;;\n   },\n   handle(handlerInput) {\n   const speakOutput = &#x27;Welcome, you can say Hello or Help. Which would you like to try?&#x27;;\n\n   return handlerInput.responseBuilder\n   .speak(speakOutput)\n   .reprompt(speakOutput)\n   .getResponse();\n   }\n};\n\nconst HelloWorldIntentHandler = {\n   canHandle(handlerInput) {\n   return Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n   &amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) === &#x27;HelloWorldIntent&#x27;;\n   },\n   handle(handlerInput) {\n   const speakOutput = &#x27;Hello World!&#x27;;\n\n   return handlerInput.responseBuilder\n   .speak(speakOutput)\n   //.reprompt(&#x27;add a reprompt if you want to keep the session open for the user to respond&#x27;)\n   .getResponse();\n   }\n};\n\nconst HelpIntentHandler = {\n   canHandle(handlerInput) {\n   return Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n   &amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) === &#x27;AMAZON.HelpIntent&#x27;;\n   },\n   handle(handlerInput) {\n   const speakOutput = &#x27;You can say hello to me! How can I help?&#x27;;\n\n   return handlerInput.responseBuilder\n   .speak(speakOutput)\n   .reprompt(speakOutput)\n   .getResponse();\n   }\n};\n\nconst

```
CancelAndStopIntentHandler = {\n    canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; (Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;AMAZON.CancelIntent&#x27;\n                ||
Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;AMAZON.StopIntent&#x27;);\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;Goodbye!&#x27;;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .getRespons
e();\n    }\n};\n/* *\n * FallbackIntent triggers when a customer says something that
doesn\u2019t map to any intents in your skill\n * It must also be defined in the
language model (if the locale supports it)\n * This handler can be safely added but
will be ingnored in locales that do not support it yet \n * */\nconst
FallbackIntentHandler = {\n    canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;AMAZON.FallbackIntent&#x27;;\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;Sorry, I don\\&#x27;t know about that. Please try
again.&#x27;;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .reprompt(s
peakOutput)\n            .getResponse();\n    }\n};\n/* *\n * SessionEndedRequest
notifies that a session was ended. This handler will be triggered when a currently
open \n * session is closed for one of the following reasons: 1) The user says
&quot;exit&quot; or &quot;quit&quot;. 2) The user does not \n * respond or says
something that does not match an intent defined in your voice model. 3) An error
occurs \n * */\nconst SessionEndedRequestHandler = {\n    canHandle(handlerInput) {\n
return Alexa.getRequestType(handlerInput.requestEnvelope) ===
&#x27;SessionEndedRequest&#x27;;\n    },\n    handle(handlerInput) {\n
console.log(`~~~~ Session ended: ${JSON.stringify(handlerInput.requestEnvelope)}`);\n
// Any cleanup logic goes here.\n        return
handlerInput.responseBuilder.getResponse(); // notice we send an empty
response\n    }\n};\n/* *\n * The intent reflector is used for interaction model
testing and debugging.\n * It will simply repeat the intent the user said. You can
create custom handlers for your intents \n * by defining them above, then also adding
them to the request handler chain below \n * */\nconst IntentReflectorHandler = {\n
canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) ===
&#x27;IntentRequest&#x27;;\n    },\n    handle(handlerInput) {\n        const
intentName = Alexa.getIntentName(handlerInput.requestEnvelope);\n        const
speakOutput = `You just triggered ${intentName}`;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n
//.reprompt(&#x27;add a reprompt if you want to keep the session open for the user to
respond&#x27;)\n            .getResponse();\n    }\n};\n/**\n * Generic error handling
to capture any syntax or routing errors. If you receive an error\n * stating the
request handler chain is not found, you have not implemented a handler for\n * the
intent being invoked or included it in the skill builder below \n * */\nconst
ErrorHandler = {\n    canHandle() {\n        return true;\n    },\n
handle(handlerInput, error) {\n        const speakOutput = &#x27;Sorry, I had trouble
doing what you asked. Please try again.&#x27;;\n        console.log(`~~~~ Error
handled: ${JSON.stringify(error)}`);\n\n        return
```

handlerInput.responseBuilder\n   .speak(speakOutput)\n   .reprompt(s
peakOutput)\n   .getResponse();\n   }\n};\n\n/**\n * This handler acts as
the entry point for your skill, routing all request and response\n * payloads to the
handlers above. Make sure any new handlers or interceptors you&#x27;ve\n * defined are
included below. The order matters - they&#x27;re processed top to bottom \n *
*/\nexports.handler = Alexa.SkillBuilders.custom()\n   .addRequestHandlers(\n
LaunchRequestHandler,\n   HelloWorldIntentHandler,\n   HelpIntentHandler,\n
CancelAndStopIntentHandler,\n   FallbackIntentHandler,\n
SessionEndedRequestHandler,\n   IntentReflectorHandler)\n   .addErrorHandlers(\n
ErrorHandler)\n   .withCustomUserAgent(&#x27;sample/hello-
world/v1.2&#x27;)\n   .lambda();\n\nwhere should I add the open api call logic stuffs?
Can you write it for me? Also I&#x27;m not sure what or if I already have an API keey
from open API, can you guide me on that also? "]}, "status": "finished_successfully",
"end_turn": null, "weight": 1.0, "metadata": {"serialization_metadata":
{"custom_symbol_offsets": []}, "request_id": "8d1acce5cb2c0d9c-MXP", "message_source":
null, "timestamp_": "absolute", "message_type": null}, "recipient": "all", "channel":
null}, "parent": "36d19160-4e73-4a52-b831-6ded4b89372b", "children": ["37e5a85d-a54b-
4d0c-95d7-4085e1ba551c"]}, "37e5a85d-a54b-4d0c-95d7-4085e1ba551c": {"id": "37e5a85d-
a54b-4d0c-95d7-4085e1ba551c", "message": {"id": "37e5a85d-a54b-4d0c-95d7-4085e1ba551c",
"author": {"role": "assistant", "name": null, "metadata": {}}, "create_time":
1728775048.43216, "update_time": null, "content": {"content_type": "text", "parts":
["You&#x27;re on the right track! Let&#x27;s walk through adding the OpenAI API call
logic to your existing Alexa skill.\n\n### Step 1: Get Your OpenAI API Key\nIf you
already have an OpenAI account (OpenAI Plus), you can generate an API key by following
these steps:\n\n1. Go to [OpenAI API](https://platform.openai.com/account/api-
keys).\n2. Log in to your account.\n3. Generate a new API key if you haven\u2019t
already. Copy it for use in the code. This key is necessary for making requests to the
OpenAI API.\n\n### Step 2: Modify Your `index.js` to Include OpenAI API
Logic\n\nYou\u2019ll need to install the `axios` library to make HTTP requests. Since
Alexa-Hosted Skills don&#x27;t allow custom packages directly, you can use the `https`
built-in library instead to make the API call.\n\nBelow is how you can modify your
code to call the OpenAI API and return a response from your Alexa skill.\n\n#### 1.
Add a new **GetAIResponseIntentHandler** for handling the user\u2019s
query.\n\n```javascript\nconst https = require(&#x27;https&#x27;);\n\n// Function to
send the query to OpenAI API and get a response\nfunction getAIResponse(query) {\n
const apiKey = &#x27;your-openai-api-key&#x27;; // Replace with your OpenAI API key\n
const options = {\n   hostname: &#x27;api.openai.com&#x27;,\n   path:
&#x27;/v1/completions&#x27;,\n   method: &#x27;POST&#x27;,\n   headers: {\n
&#x27;Authorization&#x27;: `Bearer ${apiKey}`,\n   &#x27;Content-Type&#x27;:
&#x27;application/json&#x27;\n   }\n   };\n\n   const body = JSON.stringify({\n
model: &#x27;text-davinci-003&#x27;, // Adjust if needed for other models like GPT-
4\n   prompt: query,\n   max_tokens: 100,\n   temperature:
0.7\n   });\n\n   return new Promise((resolve, reject) =&gt; {\n   const req =
https.request(options, (res) =&gt; {\n   let data = &#x27;&#x27;;\n
res.on(&#x27;data&#x27;, (chunk) =&gt; {\n   data +=
chunk;\n   });\n   res.on(&#x27;end&#x27;, () =&gt; {\n
const response = JSON.parse(data);\n
resolve(response.choices[0].text.trim());\n   });\n   });\n\n
req.on(&#x27;error&#x27;, (e) =&gt; {\n   reject(e);\n   });\n\n

req.write(body);\n   req.end();\n   });\n}\n\nconst GetAIResponseIntentHandler = {\n   canHandle(handlerInput) {\n   return Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n &amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) === &#x27;GetAIResponseIntent&#x27;;\n   },\n   async handle(handlerInput) {\n const userQuery = handlerInput.requestEnvelope.request.intent.slots.query.value;\n \n   let aiResponse = &#x27;&#x27;;\n   try {\n   aiResponse = await getAIResponse(userQuery); // Send the user query to OpenAI\n   } catch (error) {\n   console.log(`Error with OpenAI API call: ${error}`);\n aiResponse = &#x27;Sorry, I had trouble processing that.&#x27;;\n   }\n\n return handlerInput.responseBuilder\n   .speak(aiResponse)\n   .getResponse();\n   }\n};\n```\n\n#### 2. Add **GetAIResponseIntentHandler** to the list of request handlers.\n\nMake sure you include your new handler in the skill builder chain at the bottom of your `index.js` file:\n\n```javascript\nexports.handler = Alexa.SkillBuilders.custom()\n   .addRequestHandlers(\n LaunchRequestHandler,\n   HelloWorldIntentHandler,\n   HelpIntentHandler,\n CancelAndStopIntentHandler,\n   GetAIResponseIntentHandler, // Add this line\n FallbackIntentHandler,\n   SessionEndedRequestHandler,\n IntentReflectorHandler)\n   .addErrorHandlers(\n ErrorHandler)\n   .withCustomUserAgent(&#x27;sample/hello-world/v1.2&#x27;)\n   .lambda();\n```\n\n### Step 3: Update Your Interaction Model\n\nEnsure that your interaction model is configured to handle the `GetAIResponseIntent`. You already set up the **query** slot using `AMAZON.SearchQuery`, so make sure you have sample utterances like these:\n\n```json\n{\n &quot;name&quot;: &quot;GetAIResponseIntent&quot;,\n &quot;slots&quot;: [\n   {\n   &quot;name&quot;: &quot;query&quot;,\n &quot;type&quot;: &quot;AMAZON.SearchQuery&quot;\n   }\n],\n &quot;samples&quot;: [\n   &quot;What is {query}?&quot;,\n   &quot;Can you explain {query}?&quot;,\n &quot;Tell me about {query}&quot;\n]\n}\n```\n\n### Step 4: Test the Skill\n\nOnce you\u2019ve made these changes:\n1. Save and deploy the code.\n2. Go to the **Test** tab in the Alexa Developer Console, and try asking questions like:\n   - &quot;What is quantum mechanics?&quot;\n   - &quot;Tell me about black holes.&quot;\n\nIf you receive responses from OpenAI via Alexa, you\u2019re all set! You can fine-tune the prompt, temperature, and other parameters of the OpenAI request to fit your needs.\n\n### Summary:\n- You updated `index.js` to include a function (`getAIResponse`) that makes an API call to OpenAI.\n- You added a new intent handler (`GetAIResponseIntentHandler`) to handle user queries and send them to OpenAI.\n- You configured the interaction model to allow Alexa to capture the user&#x27;s query using the `AMAZON.SearchQuery` slot.\n \nLet me know how it goes or if you need further adjustments!"]}, "status": "finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id": null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o", "parent_id": "aaa29840-58ab-4121-b87c-691af8b39763", "finish_details": {"type": "stop", "stop_tokens": [200002]}, "is_complete": true, "request_id": "8d1acce5cb2c0d9c-MXP", "timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "aaa29840-58ab-4121-b87c-691af8b39763", "children": ["aaa2db05-9e02-42b7-8a5c-c92b2a35ca57"]}, "aaa2db05-9e02-42b7-8a5c-c92b2a35ca57": {"id": "aaa2db05-9e02-42b7-8a5c-c92b2a35ca57", "message": {"id": "aaa2db05-9e02-42b7-8a5c-c92b2a35ca57", "author": {"role": "user",

"name": null, "metadata": {}}, "create_time": 1728775288.748154, "update_time": null,
"content": {"content_type": "text", "parts": ["yeah but where do I place these code
snippets you gave me? also I have the api key. Can you make the place for putting the
api key in the code a simple find and replace deal for me. like make key a word
already in whatever brackets or whatever it needs to be in"]}, "status":
"finished_successfully", "end_turn": null, "weight": 1.0, "metadata":
{"serialization_metadata": {"custom_symbol_offsets": []}, "request_id":
"8d1ad38ec8c00d9c-MXP", "message_source": null, "timestamp_": "absolute",
"message_type": null}, "recipient": "all", "channel": null}, "parent": "37e5a85d-a54b-
4d0c-95d7-4085e1ba551c", "children": ["7f12d99b-ec70-4cce-9fbb-418f0846ccf2"]},
"7f12d99b-ec70-4cce-9fbb-418f0846ccf2": {"id": "7f12d99b-ec70-4cce-9fbb-418f0846ccf2",
"message": {"id": "7f12d99b-ec70-4cce-9fbb-418f0846ccf2", "author": {"role":
"assistant", "name": null, "metadata": {}}, "create_time": 1728775339.945629,
"update_time": null, "content": {"content_type": "text", "parts": ["Got it! I&#x27;ll
guide you on where exactly to place the code snippets in your existing `index.js` file.
I\u2019ll also make the API key placement as easy as a **find-and-replace**.\n\n###
Complete Breakdown:\n\n1. **Place the OpenAI API Key Placeholder**:\n   I\u2019ll add
a placeholder for your API key (`{{API_KEY}}`) so you can easily replace it.\n\n2.
Insert the OpenAI API Call Logic:\n   This will include the `getAIResponse`
function and the `GetAIResponseIntentHandler` that you will add to your current
code.\n\n3. **Ensure the Correct Placement**:\n   I\u2019ll show you exactly where to
put each part in your existing `index.js` code.\n\n### Modified `index.js` File with
Comments\n\n```javascript\nconst Alexa = require(&#x27;ask-sdk-core&#x27;);\nconst
https = require(&#x27;https&#x27;); // Required for making the OpenAI API
request\n\n// Function to send the query to OpenAI API and get a response\nfunction
getAIResponse(query) {\n   const apiKey = &#x27;{{API_KEY}}&#x27;; // &lt;&lt;&lt;
FIND AND REPLACE THIS with your actual OpenAI API key\n   const options = {\n
hostname: &#x27;api.openai.com&#x27;,\n   path: &#x27;/v1/completions&#x27;,\n
method: &#x27;POST&#x27;,\n   headers: {\n   &#x27;Authorization&#x27;:
`Bearer ${apiKey}`,\n   &#x27;Content-Type&#x27;:
&#x27;application/json&#x27;\n   }\n   };\n\n   const body = JSON.stringify({\n
model: &#x27;text-davinci-003&#x27;, // Adjust if needed for other models\n
prompt: query,\n   max_tokens: 100,\n   temperature: 0.7\n   });\n\n
return new Promise((resolve, reject) =&gt; {\n   const req =
https.request(options, (res) =&gt; {\n   let data = &#x27;&#x27;;\n
res.on(&#x27;data&#x27;, (chunk) =&gt; {\n   data +=
chunk;\n   });\n   res.on(&#x27;end&#x27;, () =&gt; {\n
const response = JSON.parse(data);\n
resolve(response.choices[0].text.trim());\n   });\n   });\n\n
req.on(&#x27;error&#x27;, (e) =&gt; {\n   reject(e);\n   });\n\n
req.write(body);\n   req.end();\n   });\n}\n\n// New Intent Handler to call
OpenAI API and get the response\nconst GetAIResponseIntentHandler = {\n
canHandle(handlerInput) {\n   return
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;GetAIResponseIntent&#x27;;\n   },\n   async handle(handlerInput) {\n
const userQuery = handlerInput.requestEnvelope.request.intent.slots.query.value;\n
\n   let aiResponse = &#x27;&#x27;;\n   try {\n   aiResponse =
await getAIResponse(userQuery); // Call the OpenAI API here\n   } catch (error)

```
{\n            console.log(`Error with OpenAI API call: ${error}`);\n
aiResponse = &#x27;Sorry, I had trouble processing that.&#x27;;\n        }\n\n
return
handlerInput.responseBuilder\n            .speak(aiResponse)\n            .getResponse
();\n    }\n};\n\nconst LaunchRequestHandler = {\n    canHandle(handlerInput) {\n
return Alexa.getRequestType(handlerInput.requestEnvelope) ===
&#x27;LaunchRequest&#x27;;\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;Welcome, you can say Hello or Help. Which would you like to
try?&#x27;;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .reprompt(s
peakOutput)\n            .getResponse();\n    }\n};\n\nconst HelloWorldIntentHandler =
{\n    canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;HelloWorldIntent&#x27;;\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;Hello World!&#x27;;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .getRespons
e();\n    }\n};\n\nconst HelpIntentHandler = {\n    canHandle(handlerInput) {\n
return Alexa.getRequestType(handlerInput.requestEnvelope) ===
&#x27;IntentRequest&#x27;\n            &amp;&amp;
Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;AMAZON.HelpIntent&#x27;;\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;You can say hello to me! How can I help?&#x27;;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .reprompt(s
peakOutput)\n            .getResponse();\n    }\n};\n\nconst
CancelAndStopIntentHandler = {\n    canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; (Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;AMAZON.CancelIntent&#x27;\n                ||
Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;AMAZON.StopIntent&#x27;);\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;Goodbye!&#x27;;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .getRespons
e();\n    }\n};\n\nconst FallbackIntentHandler = {\n    canHandle(handlerInput) {\n
return Alexa.getRequestType(handlerInput.requestEnvelope) ===
&#x27;IntentRequest&#x27;\n            &amp;&amp;
Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;AMAZON.FallbackIntent&#x27;;\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;Sorry, I don\\&#x27;t know about that. Please try
again.&#x27;;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .reprompt(s
peakOutput)\n            .getResponse();\n    }\n};\n\nconst
SessionEndedRequestHandler = {\n    canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) ===
&#x27;SessionEndedRequest&#x27;;\n    },\n    handle(handlerInput) {\n
console.log(`~~~~ Session ended: ${JSON.stringify(handlerInput.requestEnvelope)}`);\n
return handlerInput.responseBuilder.getResponse(); // empty
response\n    }\n};\n\nconst IntentReflectorHandler = {\n    canHandle(handlerInput)
{\n        return Alexa.getRequestType(handlerInput.requestEnvelope) ===
```

&#x27;IntentRequest&#x27;;\n   },\n   handle(handlerInput) {\n   const
intentName = Alexa.getIntentName(handlerInput.requestEnvelope);\n   const
speakOutput = `You just triggered ${intentName}`;\n\n   return
handlerInput.responseBuilder\n   .speak(speakOutput)\n   .getRespons
e();\n   }\n};\n\nconst ErrorHandler = {\n   canHandle() {\n   return
true;\n   },\n   handle(handlerInput, error) {\n   const speakOutput =
&#x27;Sorry, I had trouble doing what you asked. Please try again.&#x27;;\n
console.log(`~~~~ Error handled: ${JSON.stringify(error)}`);\n\n   return
handlerInput.responseBuilder\n   .speak(speakOutput)\n   .reprompt(s
peakOutput)\n   .getResponse();\n   }\n};\n\n// Add
GetAIResponseIntentHandler to the skill&#x27;s request handlers\nexports.handler =
Alexa.SkillBuilders.custom()\n   .addRequestHandlers(\n
LaunchRequestHandler,\n   HelloWorldIntentHandler,\n   HelpIntentHandler,\n
CancelAndStopIntentHandler,\n   GetAIResponseIntentHandler, // &lt;&lt;&lt; Add
this here\n   FallbackIntentHandler,\n   SessionEndedRequestHandler,\n
IntentReflectorHandler)\n   .addErrorHandlers(\n
ErrorHandler)\n   .withCustomUserAgent(&#x27;sample/hello-
world/v1.2&#x27;)\n   .lambda();\n```\n\n### Explanation:\n1. **Placeholders**: \n
- I added the placeholder `{{API_KEY}}` for your OpenAI API key. Just **find and
replace** `{{API_KEY}}` with your actual key.\n\n2. **`getAIResponse` Function**: \n
- This function is placed near the top of the file. It makes the request to the OpenAI
API when a user asks a question.\n\n3. **New `GetAIResponseIntentHandler`**: \n   -
This handler captures the user&#x27;s query, calls the OpenAI API, and returns the
result. I placed it below the existing handlers for better organization.\n\n4.
Adding the Handler to the Chain: \n   - The `GetAIResponseIntentHandler` is added
to the request handler chain near the bottom, so Alexa can handle this new
intent.\n\n### Next Steps:\n1. Replace the placeholder `{{API_KEY}}` with your actual
OpenAI API key.\n2. Save and deploy the code in the **Alexa Developer Console**.\n3.
Test it in the **Test** tab of the Alexa Developer Console by asking a question (e.g.,
&quot;What is artificial intelligence?&quot;).\n\nLet me know if you need further
adjustments or help with testing!"]}, "status": "finished_successfully","end_turn":
true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "aaa2db05-9e02-42b7-8a5c-c92b2a35ca57", "finish_details": {"type": "stop",
"stop_tokens": [200002]}, "is_complete": true, "request_id": "8d1ad38ec8c00d9c-MXP",
"timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "aaa2db05-
9e02-42b7-8a5c-c92b2a35ca57", "children": ["aaa27207-346c-4688-b785-7e712a95c3b5"]},
"aaa27207-346c-4688-b785-7e712a95c3b5": {"id": "aaa27207-346c-4688-b785-7e712a95c3b5",
"message": {"id": "aaa27207-346c-4688-b785-7e712a95c3b5", "author": {"role": "user",
"name": null, "metadata": {}}, "create_time": 1728775389.021573, "update_time": null,
"content": {"content_type": "text", "parts": ["so is the API_KEY the place holder or
do I need to replace the brackets or whatever those swirley things are too? "]},
"status": "finished_successfully", "end_turn": null, "weight": 1.0, "metadata":
{"serialization_metadata": {"custom_symbol_offsets": []}, "request_id":
"8d1ad6032f3e0d9c-MXP", "message_source": null, "timestamp_": "absolute",
"message_type": null}, "recipient": "all", "channel": null}, "parent": "7f12d99b-ec70-
4cce-9fbb-418f0846ccf2", "children": ["c0eea5db-4f2b-41e3-b786-4af54315b434"]},
"c0eea5db-4f2b-41e3-b786-4af54315b434": {"id": "c0eea5db-4f2b-41e3-b786-4af54315b434",

```
"message": {"id": "c0eea5db-4f2b-41e3-b786-4af54315b434", "author": {"role":
"assistant", "name": null, "metadata": {}}, "create_time": 1728775396.149795,
"update_time": null, "content": {"content_type": "text", "parts": ["You only need to
replace the **`API_KEY`** placeholder text. You can leave the curly braces (`{{` and
`}}`) as they are. So, when you&#x27;re replacing it, just do:\n\n```javascript\nconst
apiKey = &#x27;your-actual-api-key-here&#x27;;\n```\n\nFor
example:\n\n```javascript\nconst apiKey = &#x27;sk-12345abcd&#x27;;  // Replace this
with your real OpenAI API key\n```\n\nIn summary:\n- Replace only `{{API_KEY}}` with
your actual API key.\n- Keep everything else, including the quotes and the format,
exactly as it is.\n\nLet me know if that clears it up!"]}, "status":
"finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "aaa27207-346c-4688-b785-7e712a95c3b5", "finish_details": {"type": "stop",
"stop_tokens": [200002]}, "is_complete": true, "request_id": "8d1ad6032f3e0d9c-MXP",
"timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "aaa27207-
346c-4688-b785-7e712a95c3b5", "children": ["aaa2327c-92a7-4024-a851-7182cc81abca"]},
"aaa2327c-92a7-4024-a851-7182cc81abca": {"id": "aaa2327c-92a7-4024-a851-7182cc81abca",
"message": {"id": "aaa2327c-92a7-4024-a851-7182cc81abca", "author": {"role": "user",
"name": null, "metadata": {}}, "create_time": 1728775545.302993, "update_time": null,
"content": {"content_type": "text", "parts": ["so after which line in the origonal
node.js code I gave you below should I past the code snipet after the phrase
&#x27;file with comments&#x27;?\n\n/* *\n * This sample demonstrates handling intents
from an Alexa skill using the Alexa Skills Kit SDK (v2).\n * Please visit
https://alexa.design/cookbook for additional examples on implementing slots, dialog
management,\n * session persistence, api calls, and more.\n * */\nconst Alexa =
require(&#x27;ask-sdk-core&#x27;);\n\nconst LaunchRequestHandler = {\n
canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) ===
&#x27;LaunchRequest&#x27;;\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;Welcome, you can say Hello or Help. Which would you like to
try?&#x27;;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .reprompt(s
peakOutput)\n            .getResponse();\n    }\n};\n\nconst HelloWorldIntentHandler =
{\n    canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;HelloWorldIntent&#x27;;\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;Hello World!&#x27;;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n
//.reprompt(&#x27;add a reprompt if you want to keep the session open for the user to
respond&#x27;)\n            .getResponse();\n    }\n};\n\nconst HelpIntentHandler =
{\n    canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;AMAZON.HelpIntent&#x27;;\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;You can say hello to me! How can I help?&#x27;;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .reprompt(s
peakOutput)\n            .getResponse();\n    }\n};\n\nconst
```

```
CancelAndStopIntentHandler = {\n    canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; (Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;AMAZON.CancelIntent&#x27;\n                ||
Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;AMAZON.StopIntent&#x27;);\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;Goodbye!&#x27;;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .getRespons
e();\n    }\n};\n/* *\n * FallbackIntent triggers when a customer says something that
doesn\u2019t map to any intents in your skill\n * It must also be defined in the
language model (if the locale supports it)\n * This handler can be safely added but
will be ingnored in locales that do not support it yet \n * */\nconst
FallbackIntentHandler = {\n    canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;AMAZON.FallbackIntent&#x27;;\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;Sorry, I don\\&#x27;t know about that. Please try
again.&#x27;;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .reprompt(s
peakOutput)\n            .getResponse();\n    }\n};\n/* *\n * SessionEndedRequest
notifies that a session was ended. This handler will be triggered when a currently
open \n * session is closed for one of the following reasons: 1) The user says
&quot;exit&quot; or &quot;quit&quot;. 2) The user does not \n * respond or says
something that does not match an intent defined in your voice model. 3) An error
occurs \n * */\nconst SessionEndedRequestHandler = {\n    canHandle(handlerInput) {\n
return Alexa.getRequestType(handlerInput.requestEnvelope) ===
&#x27;SessionEndedRequest&#x27;;\n    },\n    handle(handlerInput) {\n
console.log(`~~~~ Session ended: ${JSON.stringify(handlerInput.requestEnvelope)}`);\n
// Any cleanup logic goes here.\n        return
handlerInput.responseBuilder.getResponse(); // notice we send an empty
response\n    }\n};\n/* *\n * The intent reflector is used for interaction model
testing and debugging.\n * It will simply repeat the intent the user said. You can
create custom handlers for your intents \n * by defining them above, then also adding
them to the request handler chain below \n * */\nconst IntentReflectorHandler = {\n
canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) ===
&#x27;IntentRequest&#x27;;\n    },\n    handle(handlerInput) {\n        const
intentName = Alexa.getIntentName(handlerInput.requestEnvelope);\n        const
speakOutput = `You just triggered ${intentName}`;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n
//.reprompt(&#x27;add a reprompt if you want to keep the session open for the user to
respond&#x27;)\n            .getResponse();\n    }\n};\n/**\n * Generic error handling
to capture any syntax or routing errors. If you receive an error\n * stating the
request handler chain is not found, you have not implemented a handler for\n * the
intent being invoked or included it in the skill builder below \n * */\nconst
ErrorHandler = {\n    canHandle() {\n        return true;\n    },\n
handle(handlerInput, error) {\n        const speakOutput = &#x27;Sorry, I had trouble
doing what you asked. Please try again.&#x27;;\n        console.log(`~~~~ Error
handled: ${JSON.stringify(error)}`);\n\n        return
```

~~~
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .reprompt(s
peakOutput)\n            .getResponse();\n    }\n};\n\n/**\n * This handler acts as
the entry point for your skill, routing all request and response\n * payloads to the
handlers above. Make sure any new handlers or interceptors you&#x27;ve\n * defined are
included below. The order matters - they&#x27;re processed top to bottom \n *
*/\nexports.handler = Alexa.SkillBuilders.custom()\n    .addRequestHandlers(\n
LaunchRequestHandler,\n        HelloWorldIntentHandler,\n        HelpIntentHandler,\n
CancelAndStopIntentHandler,\n        FallbackIntentHandler,\n
SessionEndedRequestHandler,\n        IntentReflectorHandler)\n    .addErrorHandlers(\n
ErrorHandler)\n    .withCustomUserAgent(&#x27;sample/hello-
world/v1.2&#x27;)\n    .lambda();"]}, "status": "finished_successfully", "end_turn":
null, "weight": 1.0, "metadata": {"serialization_metadata": {"custom_symbol_offsets":
[]}, "targeted_reply": "File with Comments", "request_id": "8d1ad9d43c490d9c-MXP",
"message_source": null, "timestamp_": "absolute", "message_type": null}, "recipient":
"all", "channel": null}, "parent": "c0eea5db-4f2b-41e3-b786-4af54315b434", "children":
["50fa4969-c0ea-4066-b2e5-09696bf95972"]}, "50fa4969-c0ea-4066-b2e5-09696bf95972":
{"id": "50fa4969-c0ea-4066-b2e5-09696bf95972", "message": {"id": "50fa4969-c0ea-4066-
b2e5-09696bf95972", "author": {"role": "system", "name": null, "metadata": {}},
"create_time": 1728775545.303478, "update_time": null, "content": {"content_type":
"text", "parts": [""]}, "status": "finished_successfully", "end_turn": null, "weight":
1.0, "metadata": {"request_id": "8d1ad9d43c490d9c-MXP", "message_source": null,
"timestamp_": "absolute", "message_type": null, "is_visually_hidden_from_conversation":
true}, "recipient": "all", "channel": null}, "parent": "aaa2327c-92a7-4024-a851-
7182cc81abca", "children": ["81d4062e-9955-4358-b88f-0c0dc2561eb2"]}, "81d4062e-9955-
4358-b88f-0c0dc2561eb2": {"id": "81d4062e-9955-4358-b88f-0c0dc2561eb2", "message":
{"id": "81d4062e-9955-4358-b88f-0c0dc2561eb2", "author": {"role": "assistant", "name":
null, "metadata": {}}, "create_time": 1728775571.358405, "update_time": null,
"content": {"content_type": "text", "parts": ["You should place the OpenAI API call
logic and the related intent handler after the `require` statements but before the
existing intent handlers. Here&#x27;s a step-by-step guide for where to insert the
code:\n\n### Step-by-Step Placement\n\n1. **Locate** the part of your code where
external libraries are required. In your case, this is near the top:\n
```javascript\n   const Alexa = require(&#x27;ask-sdk-core&#x27;);\n   ```\n\n2.
**Insert the OpenAI API logic and handler** right after the `const Alexa =
require(&#x27;ask-sdk-core&#x27;);` line. \n\n### Here&#x27;s how your file should
look:\n\n#### Insert this code snippet after the `const Alexa = require(&#x27;ask-sdk-
core&#x27;);` line:\n\n```javascript\nconst https = require(&#x27;https&#x27;);  //
Required for making the OpenAI API request\n\n// Function to send the query to OpenAI
API and get a response\nfunction getAIResponse(query) {\n    const apiKey =
&#x27;{{API_KEY}}&#x27;;  // &lt;&lt;&lt; FIND AND REPLACE THIS with your actual
OpenAI API key\n    const options = {\n        hostname: &#x27;api.openai.com&#x27;,\n
path: &#x27;/v1/completions&#x27;,\n        method: &#x27;POST&#x27;,\n        headers:
{\n            &#x27;Authorization&#x27;: `Bearer ${apiKey}`,\n
&#x27;Content-Type&#x27;: &#x27;application/json&#x27;\n        }\n    };\n\n    const
body = JSON.stringify({\n        model: &#x27;text-davinci-003&#x27;,  // Adjust if
needed for other models\n        prompt: query,\n        max_tokens: 100,\n
temperature: 0.7\n    });\n\n    return new Promise((resolve, reject) =&gt; {\n
const req = https.request(options, (res) =&gt; {\n            let data =
&#x27;&#x27;;\n            res.on(&#x27;data&#x27;, (chunk) =&gt; {\n
~~~

```
data += chunk;\n            });\n            res.on(&#x27;end&#x27;, () =&gt; {\n
const response = JSON.parse(data);\n
resolve(response.choices[0].text.trim());\n            });\n        });\n\n
req.on(&#x27;error&#x27;, (e) =&gt; {\n            reject(e);\n        });\n\n
req.write(body);\n        req.end();\n    });\n}\n\n// New Intent Handler to call
OpenAI API and get the response\nconst GetAIResponseIntentHandler = {\n
canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;GetAIResponseIntent&#x27;;\n    },\n    async handle(handlerInput) {\n
const userQuery = handlerInput.requestEnvelope.request.intent.slots.query.value;\n
\n        let aiResponse = &#x27;&#x27;;\n        try {\n            aiResponse =
await getAIResponse(userQuery);  // Call the OpenAI API here\n        } catch (error)
{\n            console.log(`Error with OpenAI API call: ${error}`);\n
aiResponse = &#x27;Sorry, I had trouble processing that.&#x27;;\n        }\n\n
return
handlerInput.responseBuilder\n            .speak(aiResponse)\n            .getResponse
();\n    }\n};\n```\n\n### Now, add the new handler to the skill handler chain:\n1.
Scroll down to the **Skill Handler Section** at the bottom of your `index.js` file.\n2.
Find the `.addRequestHandlers` method where other handlers like `LaunchRequestHandler`
are already listed.\n3. Add `GetAIResponseIntentHandler` to the list, as
shown:\n\n```javascript\nexports.handler =
Alexa.SkillBuilders.custom()\n    .addRequestHandlers(\n
LaunchRequestHandler,\n        HelloWorldIntentHandler,\n        HelpIntentHandler,\n
CancelAndStopIntentHandler,\n        FallbackIntentHandler,\n
GetAIResponseIntentHandler,  // &lt;&lt;&lt; Add this line to register the new
handler\n        SessionEndedRequestHandler,\n
IntentReflectorHandler)\n    .addErrorHandlers(\n
ErrorHandler)\n    .withCustomUserAgent(&#x27;sample/hello-
world/v1.2&#x27;)\n    .lambda();\n```\n\n### Recap:\n1. Add the **OpenAI logic** and
**`GetAIResponseIntentHandler`** directly after your `require` statements.\n2. Add
**`GetAIResponseIntentHandler`** to the **`.addRequestHandlers()`** list near the
bottom of your file.\n3. Don&#x27;t forget to **replace** `{{API_KEY}}` with your
actual OpenAI API key.\n\nOnce you do this, save the file and test it. Let me know if
you run into any issues!"]}, "status": "finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "50fa4969-c0ea-4066-b2e5-09696bf95972", "finish_details": {"type": "stop",
"stop_tokens": [200002]}, "is_complete": true, "request_id": "8d1ad9d43c490d9c-MXP",
"timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "50fa4969-
c0ea-4066-b2e5-09696bf95972", "children": ["aaa24a51-eebc-4b4a-b86e-7c161598b57a",
"aaa26934-4f4c-409a-892e-a7b183886f73"]}, "aaa24a51-eebc-4b4a-b86e-7c161598b57a":
{"id": "aaa24a51-eebc-4b4a-b86e-7c161598b57a", "message": {"id": "aaa24a51-eebc-4b4a-
b86e-7c161598b57a", "author": {"role": "user", "name": null, "metadata": {}},
"create_time": 1728777159.802424, "update_time": null, "content": {"content_type":
"text", "parts": ["GOT THIS ERROR WHEN TRYING TO BUILD. WHAT SHOULD I DO? "]},
"status": "finished_successfully", "end_turn": null, "weight": 1.0, "metadata":
{"serialization_metadata": {"custom_symbol_offsets": []}, "request_id":
"8d1b013bdba60e5e-MXP", "message_source": null, "timestamp_": "absolute",
```

"message_type": null}, "recipient": "all", "channel": null}, "parent": "81d4062e-9955-4358-b88f-0c0dc2561eb2", "children": ["8e7b178d-b88d-4af9-976e-9e2ae25d6c15"]}, "8e7b178d-b88d-4af9-976e-9e2ae25d6c15": {"id": "8e7b178d-b88d-4af9-976e-9e2ae25d6c15", "message": {"id": "8e7b178d-b88d-4af9-976e-9e2ae25d6c15", "author": {"role": "assistant", "name": null, "metadata": {}}, "create_time": 1728777180.485944, "update_time": null, "content": {"content_type": "text", "parts": ["It sounds like there might be an issue during the build process in the Alexa Developer Console. To help you better, can you provide the exact error message you received? The error message will give me more details to pinpoint the issue and provide a solution.\n\nIn the meantime, here are a few **common issues and solutions** that might help:\n\n### 1. **Missing/Incorrect API Key**:\n   - Make sure you have replaced `{{API_KEY}}` with your actual OpenAI API key.\n   - The correct line should look like:\n   ```javascript\n   const apiKey = &#x27;your-actual-api-key-here&#x27;;\n   ```\n   - If the API key is missing or incorrect, it could prevent the skill from successfully calling the OpenAI API.\n\n### 2. **Incorrect Handler Registration**:\n   - Double-check that you have added the `GetAIResponseIntentHandler` correctly in the `.addRequestHandlers()` section, as well as defined it above. Make sure it looks like this at the bottom of your code:\n   ```javascript\n   exports.handler = Alexa.SkillBuilders.custom()\n   .addRequestHandlers(\n   LaunchRequestHandler,\n   HelloWorldIntentHandler,\n   HelpIntentHandler,\n   CancelAndStopIntentHandler,\n   FallbackIntentHandler,\n   GetAIResponseIntentHandler, // &lt;&lt;&lt; Ensure this line is here\n   SessionEndedRequestHandler,\n   IntentReflectorHandler)\n   .addErrorHandlers(\n   ErrorHandler)\n   .withCustomUserAgent(&#x27;sample/hello-world/v1.2&#x27;)\n   .lambda();\n   ```\n\n### 3. **JSON Parsing Issues**:\n   - If the build error mentions something like &quot;Unexpected token&quot; or &quot;Invalid JSON,&quot; there might be a formatting issue (extra commas, missing brackets) in the JSON part of the request body you\u2019re sending to OpenAI.\n   - Ensure that your `JSON.stringify()` in the `getAIResponse` function is properly formatted like this:\n   ```javascript\n   const body = JSON.stringify({\n   model: &#x27;text-davinci-003&#x27;,\n   prompt: query,\n   max_tokens: 100,\n   temperature: 0.7\n   });\n   ```\n\n### 4. **Network Errors**:\n   - If the error is related to network or SSL (like connecting to the OpenAI API), make sure you&#x27;re calling OpenAI&#x27;s API over HTTPS, which is correctly set in your code:\n   ```javascript\n   const options = {\n   hostname: &#x27;api.openai.com&#x27;,\n   path: &#x27;/v1/completions&#x27;,\n   method: &#x27;POST&#x27;,\n   headers: {\n   &#x27;Authorization&#x27;: `Bearer ${apiKey}`,\n   &#x27;Content-Type&#x27;: &#x27;application/json&#x27;\n   }\n   };\n   ```\n\n### 5. **Lambda Function Timeout**:\n   - If OpenAI API responses are slow, you might encounter timeouts. Try reducing the response size or setting a larger timeout in Alexa Developer Console if possible.\n\n### Next Steps:\n1. Please provide the **exact error message** you&#x27;re seeing, and I can give more precise guidance.\n2. If you&#x27;re stuck on a specific issue (syntax, API call), feel free to describe it, and I can help you troubleshoot.\n\nLet me know the details of the error so I can guide you further!"]}, "status": "finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id": null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",

```
"parent_id": "aaa24a51-eebc-4b4a-b86e-7c161598b57a", "finish_details": {"type": "stop",
"stop_tokens": [200002]}, "is_complete": true, "request_id": "8d1b013bdba60e5e-MXP",
"timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "aaa24a51-
eebc-4b4a-b86e-7c161598b57a", "children": []}, "aaa26934-4f4c-409a-892e-a7b183886f73":
{"id": "aaa26934-4f4c-409a-892e-a7b183886f73", "message": {"id": "aaa26934-4f4c-409a-
892e-a7b183886f73", "author": {"role": "user", "name": null, "metadata": {}},
"create_time": 1728777183.941712, "update_time": null, "content": {"content_type":
"text", "parts": ["GOT THIS ERROR WHEN TRYING TO BUILD: At least one prompt is
required for intent &quot;GetAIResponseIntent&quot; as it requires confirmation. WHAT
SHOULD I DO? "]}, "status": "finished_successfully", "end_turn": null, "weight": 1.0,
"metadata": {"request_id": "8d1b01d57f940e5e-MXP", "message_source": null,
"timestamp_": "absolute", "message_type": null}, "recipient": "all", "channel": null},
"parent": "81d4062e-9955-4358-b88f-0c0dc2561eb2", "children": ["75b8c7f4-c3c6-4735-
8229-9eaf3eb139dd"]}, "75b8c7f4-c3c6-4735-8229-9eaf3eb139dd": {"id": "75b8c7f4-c3c6-
4735-8229-9eaf3eb139dd", "message": {"id": "75b8c7f4-c3c6-4735-8229-9eaf3eb139dd",
"author": {"role": "assistant", "name": null, "metadata": {}}, "create_time":
1728777196.752839, "update_time": null, "content": {"content_type": "text", "parts":
["The error you&#x27;re seeing is because your `GetAIResponseIntent` in the
interaction model needs at least one **sample utterance** (also known as a
&quot;prompt&quot;) so that Alexa knows what to listen for when the user triggers this
intent.\n\n### Here&#x27;s how you can fix it:\n\n1. **Go to the Alexa Developer
Console**:\n   - Navigate to the **Interaction Model** for your skill.\n\n2. **Add
Sample Utterances** for the `GetAIResponseIntent`:\n   - You need to define phrases
that a user might say to trigger this intent. Since you&#x27;re using a slot called
`query`, you\u2019ll provide some sample utterances that include this slot.\n   -
Sample utterances could be like:\n     - &quot;What is {query}?&quot;\n     -
&quot;Tell me about {query}.&quot;\n     - &quot;Explain {query} to me.&quot;\n     -
&quot;Can you help me with {query}?&quot;\n\n3. **Update the Interaction Model**:\n
- In the Interaction Model section of the Developer Console:\n     1. Go to
**Intents**.\n     2. Find and click on **GetAIResponseIntent**.\n     3. Add several
**Sample Utterances** that include the **`{query}`** slot, like the examples
above.\n\n4. **Save and Build** the Interaction Model:\n   - After adding the
utterances, click **Save** and then **Build Model**. This will update the interaction
model so that Alexa can recognize when the user is trying to ask a question using this
intent.\n\n### Example of how it should look:\n\n```json\n{\n  &quot;name&quot;:
&quot;GetAIResponseIntent&quot;,\n  &quot;slots&quot;: [\n    {\n
&quot;name&quot;: &quot;query&quot;,\n      &quot;type&quot;:
&quot;AMAZON.SearchQuery&quot;\n    }\n  ],\n  &quot;samples&quot;: [\n    &quot;What
is {query}?&quot;,\n    &quot;Tell me about {query}&quot;,\n    &quot;Explain
{query}&quot;,\n    &quot;Can you help me with {query}?&quot;\n  ]\n}\n```\n\n###
Recap:\n1. Go to the **Interaction Model** tab in the Alexa Developer Console.\n2.
Find the **GetAIResponseIntent**.\n3. Add at least **four sample utterances** that
include the `{query}` slot.\n4. **Save and Build** your model.\n\nOnce you&#x27;ve
done that, the error should go away. Let me know how it goes!"]}, "status":
"finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "finish_details": {"type": "stop", "stop_tokens": [200002]}, "is_complete": true,
"message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "aaa26934-4f4c-409a-892e-a7b183886f73", "request_id": "8d1b01d57f940e5e-
```

MXP", "timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent":
"aaa26934-4f4c-409a-892e-a7b183886f73", "children": ["aaa2c5e0-dd47-4177-9497-
44a3b9f292d6"]}, "aaa2c5e0-dd47-4177-9497-44a3b9f292d6": {"id": "aaa2c5e0-dd47-4177-
9497-44a3b9f292d6", "message": {"id": "aaa2c5e0-dd47-4177-9497-44a3b9f292d6", "author":
{"role": "user", "name": null, "metadata": {}}, "create_time": 1728777231.287466,
"update_time": null, "content": {"content_type": "text", "parts": [" I ALREADY HAVE
THAT"]}, "status": "finished_successfully", "end_turn": null, "weight": 1.0,
"metadata": {"serialization_metadata": {"custom_symbol_offsets": []}, "request_id":
"8d1b02fdadcd0e5e-MXP", "message_source": null, "timestamp_": "absolute",
"message_type": null}, "recipient": "all", "channel": null}, "parent": "75b8c7f4-c3c6-
4735-8229-9eaf3eb139dd", "children": ["28b14262-e6a5-4036-b580-f91541ef08a2"]},
"28b14262-e6a5-4036-b580-f91541ef08a2": {"id": "28b14262-e6a5-4036-b580-f91541ef08a2",
"message": {"id": "28b14262-e6a5-4036-b580-f91541ef08a2", "author": {"role":
"assistant", "name": null, "metadata": {}}, "create_time": 1728777242.490289,
"update_time": null, "content": {"content_type": "text", "parts": ["If you already
have sample utterances set up for the `GetAIResponseIntent` and you&#x27;re still
seeing the error about a required confirmation, it likely means that the intent might
have **confirmation prompts** enabled by accident or that the slot requires
confirmation.\n\n### Steps to fix the issue:\n\n1. **Check for Slot Confirmation
Requirement**:\n   - In the **Alexa Developer Console**, navigate to your
GetAIResponseIntent.\n   - Find the **`query`** slot and see if it has
&quot;Confirmation&quot; enabled. If it does, Alexa is expecting the user to confirm
their input before proceeding, which may cause the error.\n\n   To fix this:\n   - Set
Confirmation to **None** for the slot.\n   \n   Here&#x27;s how:\n   - Go to
Intents -&gt; **GetAIResponseIntent**.\n   - Find the **`query`** slot, click on
it, and look for **Slot Confirmation**.\n   - Set it to **None** (if it\u2019s
enabled).\n\n2. **Check for Intent Confirmation Requirement**:\n   - Similarly, check
if the **intent itself** requires confirmation.\n   - In the **GetAIResponseIntent**,
see if there\u2019s a setting for **Intent Confirmation**.\n   - If this is enabled,
you need to set it to **None**.\n\n3. **Save and Rebuild the Model**:\n   - After
ensuring that there are no confirmation prompts for both the slot and the intent,
save the changes and **rebuild** the model.\n\n### Recap:\n- Go to your
`query` slot in the `GetAIResponseIntent` and disable **Slot Confirmation** if
it\u2019s enabled.\n- Check for any **Intent Confirmation** settings and disable
them.\n- Save and build your model again.\n\nLet me know if this resolves the
issue!"]}, "status": "finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "aaa2c5e0-dd47-4177-9497-44a3b9f292d6", "finish_details": {"type": "stop",
"stop_tokens": [200002]}, "is_complete": true, "request_id": "8d1b02fdadcd0e5e-MXP",
"timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "aaa2c5e0-
dd47-4177-9497-44a3b9f292d6", "children": ["aaa2ccb7-0639-49cf-ac1c-df560a8578c0"]},
"aaa2ccb7-0639-49cf-ac1c-df560a8578c0": {"id": "aaa2ccb7-0639-49cf-ac1c-df560a8578c0",
"message": {"id": "aaa2ccb7-0639-49cf-ac1c-df560a8578c0", "author": {"role": "user",
"name": null, "metadata": {}}, "create_time": 1728779395.849049, "update_time": null,
"content": {"content_type": "text", "parts": ["still erroring out. is there anything
in the code about confirmations? Can we modify it on a code level: \n\n/* *\n * This
sample demonstrates handling intents from an Alexa skill using the Alexa Skills Kit
SDK (v2).\n * Please visit https://alexa.design/cookbook for additional examples on

```
implementing slots, dialog management,\n * session persistence, api calls, and more.\n
* */\nconst Alexa = require(&#x27;ask-sdk-core&#x27;);\nconst https =
require(&#x27;https&#x27;);  // Required for making the OpenAI API request\n\n//
Function to send the query to OpenAI API and get a response\nfunction
getAIResponse(query) {\n    const apiKey = &#x27;{{sk-proj-
bXGBo8nQV9Fij1AKg54pPGWgwp2E0QENPBonydsLFwQ5WsGaipn5tsKqqunCoCCFK6b21n38N6T3BlbkFJAvD0
VF30JxG47HmdDBOmsTfHVm-eHsxrYbDYluYxXyR1Bv4D40RJglPfE5X58Ea1bzQSSW0QoA}}&#x27;;  //
&lt;&lt;&lt; FIND AND REPLACE THIS with your actual OpenAI API key\n    const options
= {\n        hostname: &#x27;api.openai.com&#x27;,\n        path:
&#x27;/v1/completions&#x27;,\n        method: &#x27;POST&#x27;,\n        headers: {\n
&#x27;Authorization&#x27;: `Bearer ${apiKey}`,\n            &#x27;Content-Type&#x27;:
&#x27;application/json&#x27;\n        }\n    };\n\n    const body = JSON.stringify({\n
model: &#x27;text-davinci-003&#x27;,  // Adjust if needed for other models\n
prompt: query,\n        max_tokens: 100,\n        temperature: 0.7\n    });\n\n
return new Promise((resolve, reject) =&gt; {\n        const req =
https.request(options, (res) =&gt; {\n            let data = &#x27;&#x27;;\n
res.on(&#x27;data&#x27;, (chunk) =&gt; {\n                data +=
chunk;\n            });\n            res.on(&#x27;end&#x27;, () =&gt; {\n
const response = JSON.parse(data);\n
resolve(response.choices[0].text.trim());\n            });\n        });\n\n
req.on(&#x27;error&#x27;, (e) =&gt; {\n            reject(e);\n        });\n\n
req.write(body);\n        req.end();\n    });\n}\n\n// New Intent Handler to call
OpenAI API and get the response\nconst GetAIResponseIntentHandler = {\n
canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;GetAIResponseIntent&#x27;;\n    },\n    async handle(handlerInput) {\n
const userQuery = handlerInput.requestEnvelope.request.intent.slots.query.value;\n
\n        let aiResponse = &#x27;&#x27;;\n        try {\n            aiResponse =
await getAIResponse(userQuery);  // Call the OpenAI API here\n        } catch (error)
{\n            console.log(`Error with OpenAI API call: ${error}`);\n
aiResponse = &#x27;Sorry, I had trouble processing that.&#x27;;\n        }\n\n
return
handlerInput.responseBuilder\n            .speak(aiResponse)\n            .getResponse
();\n    }\n};\n\n\nconst LaunchRequestHandler = {\n    canHandle(handlerInput) {\n
return Alexa.getRequestType(handlerInput.requestEnvelope) ===
&#x27;LaunchRequest&#x27;;\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;Welcome, you can say Hello or Help. Which would you like to
try?&#x27;;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .reprompt(s
peakOutput)\n            .getResponse();\n    }\n};\n\nconst HelloWorldIntentHandler =
{\n    canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;HelloWorldIntent&#x27;;\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;Hello World!&#x27;;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n
//.reprompt(&#x27;add a reprompt if you want to keep the session open for the user to
respond&#x27;)\n            .getResponse();\n    }\n};\n\nconst HelpIntentHandler =
{\n    canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;AMAZON.HelpIntent&#x27;;\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;You can say hello to me! How can I help?&#x27;;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .reprompt(s
peakOutput)\n            .getResponse();\n    }\n};\n\nconst
CancelAndStopIntentHandler = {\n    canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; (Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;AMAZON.CancelIntent&#x27;\n                ||
Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;AMAZON.StopIntent&#x27;);\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;Goodbye!&#x27;;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .getRespons
e();\n    }\n};\n/* *\n * FallbackIntent triggers when a customer says something that
doesn\u2019t map to any intents in your skill\n * It must also be defined in the
language model (if the locale supports it)\n * This handler can be safely added but
will be ingnored in locales that do not support it yet \n * */\nconst
FallbackIntentHandler = {\n    canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;AMAZON.FallbackIntent&#x27;;\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;Sorry, I don\\&#x27;t know about that. Please try
again.&#x27;;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .reprompt(s
peakOutput)\n            .getResponse();\n    }\n};\n/* *\n * SessionEndedRequest
notifies that a session was ended. This handler will be triggered when a currently
open \n * session is closed for one of the following reasons: 1) The user says
&quot;exit&quot; or &quot;quit&quot;. 2) The user does not \n * respond or says
something that does not match an intent defined in your voice model. 3) An error
occurs \n * */\nconst SessionEndedRequestHandler = {\n    canHandle(handlerInput) {\n
return Alexa.getRequestType(handlerInput.requestEnvelope) ===
&#x27;SessionEndedRequest&#x27;;\n    },\n    handle(handlerInput) {\n
console.log(`~~~~ Session ended: ${JSON.stringify(handlerInput.requestEnvelope)}`);\n
// Any cleanup logic goes here.\n        return
handlerInput.responseBuilder.getResponse(); // notice we send an empty
response\n    }\n};\n/* *\n * The intent reflector is used for interaction model
testing and debugging.\n * It will simply repeat the intent the user said. You can
create custom handlers for your intents \n * by defining them above, then also adding
them to the request handler chain below \n * */\nconst IntentReflectorHandler = {\n
canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) ===
&#x27;IntentRequest&#x27;;\n    },\n    handle(handlerInput) {\n        const
intentName = Alexa.getIntentName(handlerInput.requestEnvelope);\n        const
speakOutput = `You just triggered ${intentName}`;\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n
//.reprompt(&#x27;add a reprompt if you want to keep the session open for the user to
respond&#x27;)\n            .getResponse();\n    }\n};\n/**\n * Generic error handling
```

```
to capture any syntax or routing errors. If you receive an error\n * stating the
request handler chain is not found, you have not implemented a handler for\n * the
intent being invoked or included it in the skill builder below \n * */\nconst
ErrorHandler = {\n    canHandle() {\n        return true;\n    },\n
handle(handlerInput, error) {\n        const speakOutput = &#x27;Sorry, I had trouble
doing what you asked. Please try again.&#x27;;\n        console.log(`~~~~ Error
handled: ${JSON.stringify(error)}`);\n\n        return
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .reprompt(s
peakOutput)\n            .getResponse();\n    }\n};\n\n/**\n * This handler acts as
the entry point for your skill, routing all request and response\n * payloads to the
handlers above. Make sure any new handlers or interceptors you&#x27;ve\n * defined are
included below. The order matters - they&#x27;re processed top to bottom \n *
*/\nexports.handler = Alexa.SkillBuilders.custom()\n    .addRequestHandlers(\n
LaunchRequestHandler,\n        HelloWorldIntentHandler,\n        HelpIntentHandler,\n
CancelAndStopIntentHandler,\n        FallbackIntentHandler,\n
GetAIResponseIntentHandler, \n        SessionEndedRequestHandler,\n
IntentReflectorHandler)\n    .addErrorHandlers(\n
ErrorHandler)\n    .withCustomUserAgent(&#x27;sample/hello-
world/v1.2&#x27;)\n    .lambda();"]}, "status": "finished_successfully", "end_turn":
null, "weight": 1.0, "metadata": {"serialization_metadata": {"custom_symbol_offsets":
[]}, "request_id": "8d1b37d5dbae0e53-MXP", "message_source": null, "timestamp_":
"absolute", "message_type": null}, "recipient": "all", "channel": null}, "parent":
"28b14262-e6a5-4036-b580-f91541ef08a2", "children": ["259a6895-0b78-49a8-98f8-
c8c18015566d"]}, "259a6895-0b78-49a8-98f8-c8c18015566d": {"id": "259a6895-0b78-49a8-
98f8-c8c18015566d", "message": {"id": "259a6895-0b78-49a8-98f8-c8c18015566d", "author":
{"role": "assistant", "name": null, "metadata": {}}, "create_time": 1728779420.063457,
"update_time": null, "content": {"content_type": "text", "parts": ["It looks like the
issue isn&#x27;t directly related to the code but rather the interaction model
configuration within the Alexa Developer Console. Let\u2019s try some things to
resolve the issue:\n\n### Modifying the Code:\n\nThe code you provided doesn&#x27;t
seem to have any settings for **confirmation** prompts, so there\u2019s nothing in the
code requiring confirmation. However, let\u2019s ensure that the intent handler for
`GetAIResponseIntent` doesn&#x27;t require confirmation.\n\nIf you want to forcefully
bypass any confirmation or check whether the slot requires confirmation, you can
modify your code to ensure Alexa skips any potential confirmation step.\n\n### Example
Bypassing Confirmation in the Code:\nAdd this line in the `GetAIResponseIntentHandler`
to make sure Alexa skips any slot confirmation step, in case Alexa is requiring
it.\n\n```javascript\nconst GetAIResponseIntentHandler = {\n    canHandle(handlerInput)
{\n        return Alexa.getRequestType(handlerInput.requestEnvelope) ===
&#x27;IntentRequest&#x27;\n            &amp;&amp;
Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;GetAIResponseIntent&#x27;;\n    },\n    async handle(handlerInput) {\n
const userQuery = handlerInput.requestEnvelope.request.intent.slots.query.value;\n\n
// Forcefully bypass slot confirmation, if it&#x27;s happening\n
handlerInput.requestEnvelope.request.intent.confirmationStatus = &#x27;NONE&#x27;;\n
\n        let aiResponse = &#x27;&#x27;;\n        try {\n            aiResponse =
await getAIResponse(userQuery);  // Call the OpenAI API here\n        } catch (error)
{\n            console.log(`Error with OpenAI API call: ${error}`);\n
aiResponse = &#x27;Sorry, I had trouble processing that.&#x27;;\n        }\n\n
```

return
handlerInput.responseBuilder\n   .speak(aiResponse)\n   .getResponse
();\n   }\n};\n```\n\n### Confirm Slot and Intent Configuration in Alexa Developer
Console:\n1. **Go to the Alexa Developer Console**.\n2. Open your skill, go to the
Build section, and select **Interaction Model**.\n3. **Check the Slot Type** for
your `query` slot:\n   - Ensure that the **`query` slot** is set to
`AMAZON.SearchQuery` (which you likely already did).\n4. **Check for Slot
Confirmation**:\n   - Click on the **`query` slot** inside your
`GetAIResponseIntent`.\n   - Look for any **Slot Confirmation** settings and make
sure it\u2019s set to **None**.\n\n### Confirm Intent Configuration:\n1. **Check
Intent Confirmation**:\n   - Go to **Intents**, then click on the
`GetAIResponseIntent`.\n   - Look for **Intent Confirmation** and ensure it is set to
None.\n   \n### If None of These Work:\n\nSometimes Alexa\u2019s error messaging
is tricky, and rebuilding the model after checking the slots can help clear up
persistent issues.\n\n- **Try removing the slot entirely** from the
`GetAIResponseIntent` (temporarily) to see if the intent works without it. This will
help you confirm whether the problem is with the slot configuration or elsewhere.\n-
Once it works, you can add the slot back and reconfigure its type and confirmation
status.\n\n### Recap:\n1. Make sure there\u2019s no **slot confirmation** or **intent
confirmation** in your intent\u2019s setup.\n2. Bypass confirmation in the code by
explicitly setting the `confirmationStatus` to `&#x27;NONE&#x27;`.\n3. Rebuild the
interaction model and see if the issue persists.\n\nLet me know if that helps or if
you\u2019re still running into issues!"]}, "status":
"finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "aaa2ccb7-0639-49cf-ac1c-df560a8578c0", "finish_details": {"type": "stop",
"stop_tokens": [200002]}, "is_complete": true, "request_id": "8d1b37d5dbae0e53-MXP",
"timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "aaa2ccb7-
0639-49cf-ac1c-df560a8578c0", "children": ["aaa282be-1ecc-4604-8ef8-b9eabe729aef"]},
"aaa282be-1ecc-4604-8ef8-b9eabe729aef": {"id": "aaa282be-1ecc-4604-8ef8-b9eabe729aef",
"message": {"id": "aaa282be-1ecc-4604-8ef8-b9eabe729aef", "author": {"role": "user",
"name": null, "metadata": {}}, "create_time": 1728779942.371707, "update_time": null,
"content": {"content_type": "text", "parts": ["okay i fixed that but it keeps not
using my skill i don&#x27;t think because it comes back with info from websites"]},
"status": "finished_successfully", "end_turn": null, "weight": 1.0, "metadata":
{"serialization_metadata": {"custom_symbol_offsets": []}, "request_id":
"8d1b452d0d634bf3-MXP", "message_source": null, "timestamp_": "absolute",
"message_type": null}, "recipient": "all", "channel": null}, "parent": "259a6895-0b78-
49a8-98f8-c8c18015566d", "children": ["f55e9fa0-4681-4608-96db-cc6969fcf72e"]},
"f55e9fa0-4681-4608-96db-cc6969fcf72e": {"id": "f55e9fa0-4681-4608-96db-cc6969fcf72e",
"message": {"id": "f55e9fa0-4681-4608-96db-cc6969fcf72e", "author": {"role":
"assistant", "name": null, "metadata": {}}, "create_time": 1728779970.890078,
"update_time": null, "content": {"content_type": "text", "parts": ["It sounds like
Alexa might not be triggering **your custom skill** but instead defaulting to its
built-in knowledge base to answer questions. This can happen if Alexa isn&#x27;t
recognizing the invocation phrases or intents properly. Let\u2019s make sure your
skill is invoked correctly and consistently.\n\n### Steps to Ensure Alexa Uses Your
Custom Skill:\n\n#### 1. **Ensure Your Invocation Name is Correct**\n- **Invocation

Name**: The name you use to trigger your skill must be clear and unique. This is
typically set when you create the skill and can be checked/updated in the **Build &gt;
Invocation Name** section of the Alexa Developer Console.\n \n- Make sure you\u2019re
using this **exact invocation name** when talking to Alexa. For example:\n -
&quot;Alexa, open **Smart AI Assistant**&quot;\n - &quot;Alexa, ask **Smart AI
Assistant** to answer my question&quot;\n\n#### 2. **Explicitly Invoke Your Skill**\n-
Instead of just asking the question directly, try explicitly calling your skill by
name. For example:\n - &quot;Alexa, ask **Smart AI Assistant** what is quantum
mechanics?&quot;\n - &quot;Alexa, open **Smart AI Assistant** and ask it to explain
artificial intelligence.&quot;\n\nThis forces Alexa to use your skill, bypassing its
built-in knowledge system.\n\n#### 3. **Check for the `GetAIResponseIntent`
Invocation**\n- When using your skill, make sure you\u2019re using one of the sample
utterances you defined for the `GetAIResponseIntent`. Examples might include:\n -
&quot;What is {query}?&quot;\n - &quot;Tell me about {query}.&quot;\n\nAlexa will
only route your queries to your skill if they match the sample utterances you have
defined in your interaction model.\n\n#### 4. **Test in the Alexa Developer
Console**\n- Use the **Test** tab in the Alexa Developer Console to simulate requests
and see if your skill is being invoked. You can type or speak requests to see which
intents are triggered.\n- If your skill isn\u2019t being triggered, check the JSON
request in the test tab to see if Alexa is misinterpreting the invocation or
query.\n\n#### 5. **Skill Publishing and Invocation Name Conflicts**\n- If your skill
has not been published yet, make sure you\u2019re using it in **developer mode** with
your Alexa device that is linked to your developer account.\n- If your invocation name
is too generic (e.g., &quot;AI Assistant&quot;), Alexa might be confused and try using
another skill or its own knowledge. You can try giving your skill a more unique
name.\n\n#### 6. **Add Fallback Prompt for Alexa to Guide the User**\nTo ensure Alexa
correctly asks for clarification, you can add a fallback prompt to guide the user to
the correct invocation. Here&#x27;s how you can modify the code to handle an
unexpected response:\n\n```javascript\nconst GetAIResponseIntentHandler = {\n
canHandle(handlerInput) {\n   return
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;GetAIResponseIntent&#x27;;\n   },\n   async handle(handlerInput) {\n
const userQuery = handlerInput.requestEnvelope.request.intent.slots.query.value;\n
\n   if (!userQuery) {\n   const speakOutput = &quot;I didn\u2019t
understand that. Can you please ask me a question like, &#x27;Tell me about quantum
mechanics?&#x27;&quot;;\n   return
handlerInput.responseBuilder\n   .speak(speakOutput)\n   .re
prompt(speakOutput)\n   .getResponse();\n   }\n\n   let
aiResponse = &#x27;&#x27;;\n   try {\n   aiResponse = await
getAIResponse(userQuery); // Call the OpenAI API here\n   } catch (error) {\n
console.log(`Error with OpenAI API call: ${error}`);\n   aiResponse =
&#x27;Sorry, I had trouble processing that.&#x27;;\n   }\n\n   return
handlerInput.responseBuilder\n   .speak(aiResponse)\n   .getResponse
();\n   }\n};\n```\n\nThis fallback response will help guide the user in case Alexa
doesn\u2019t properly handle the intent.\n\n### Recap:\n1. **Check your invocation
name** and make sure you&#x27;re using it correctly.\n2. **Explicitly invoke the
skill** by saying &quot;Alexa, ask [skill name]...&quot; to bypass Alexa&#x27;s
default knowledge base.\n3. Test your skill in the **Alexa Developer Console** to

```
confirm it\u2019s working as intended.\n4. **Check for conflicts** in the invocation
name or generic phrases that might confuse Alexa.\n\nLet me know if this helps or if
you need more assistance!"]}, "status": "finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "aaa282be-1ecc-4604-8ef8-b9eabe729aef", "finish_details": {"type": "stop",
"stop_tokens": [200002]}, "is_complete": true, "request_id": "8d1b452d0d634bf3-MXP",
"timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "aaa282be-
1ecc-4604-8ef8-b9eabe729aef", "children": ["aaa2896d-5aa1-43be-81d9-9d468b149fb9"]},
"aaa2896d-5aa1-43be-81d9-9d468b149fb9": {"id": "aaa2896d-5aa1-43be-81d9-9d468b149fb9",
"message": {"id": "aaa2896d-5aa1-43be-81d9-9d468b149fb9", "author": {"role": "user",
"name": null, "metadata": {}}, "create_time": 1728780060.117829, "update_time": null,
"content": {"content_type": "text", "parts": ["could the dialogue delegations strategy
be the issue?"]}, "status": "finished_successfully", "end_turn": null, "weight": 1.0,
"metadata": {"serialization_metadata": {"custom_symbol_offsets": []}, "request_id":
"8d1b480d4eee4bf3-MXP", "message_source": null, "timestamp_": "absolute",
"message_type": null}, "recipient": "all", "channel": null}, "parent": "f55e9fa0-4681-
4608-96db-cc6969fcf72e", "children": ["2b2c0aca-135e-4f90-911d-dce0360bd150"]},
"2b2c0aca-135e-4f90-911d-dce0360bd150": {"id": "2b2c0aca-135e-4f90-911d-dce0360bd150",
"message": {"id": "2b2c0aca-135e-4f90-911d-dce0360bd150", "author": {"role":
"assistant", "name": null, "metadata": {}}, "create_time": 1728780088.216936,
"update_time": null, "content": {"content_type": "text", "parts": ["Yes, the **dialog
delegation strategy** could indeed be part of the issue if it&#x27;s affecting how
Alexa handles user input and routes intents. By default, Alexa uses dialog delegation
to handle slot filling, confirmation, and intent routing. This can sometimes cause
Alexa to not pass control to your custom intent as expected, especially if
there&#x27;s any confusion or if the interaction model isn\u2019t explicitly
configured to bypass delegation.\n\nLet\u2019s review what the **dialog delegation
strategy** is and how you can control or bypass it to ensure your skill behaves as
expected.\n\n### What is Dialog Delegation Strategy?\nDialog delegation allows Alexa
to handle slot filling or confirmation automatically without sending every interaction
back to your skill&#x27;s backend for processing. This can be useful, but in your case,
it might cause Alexa to incorrectly process or delay the user\u2019s query.\n\n###
Possible Dialog Delegation Issues:\n- **Alexa attempting to confirm slot values**: If
there\u2019s a slot (e.g., `query`) that is unclear or empty, Alexa may prompt the
user again to confirm or fill in the value instead of passing the query to your
API.\n- **Alexa routing control**: If Alexa believes it can handle the dialog (like
asking for more information from the user), it may not trigger your intent as
expected.\n\n### Disabling or Modifying Dialog Delegation:\n\nTo ensure Alexa always
passes control back to your skill without automatic slot handling, you can **disable
dialog management** by ensuring you aren\u2019t using dialog directives like
`Dialog.Delegate`.\n\nHere\u2019s how to adjust your handler to prevent Alexa from
automatically handling dialog:\n\n#### 1. **Ensure No Dialog Delegation is Used in
Your Code**\nIn your current handler for `GetAIResponseIntent`, you don\u2019t seem to
be explicitly using any dialog directives. However, to ensure Alexa isn&#x27;t trying
to handle dialog behind the scenes, you can check and **disable dialog management**
explicitly.\n\nModify the `GetAIResponseIntentHandler` to **disable dialog
delegation**:\n\n```javascript\nconst GetAIResponseIntentHandler = {\n
canHandle(handlerInput) {\n        return
```

```
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;GetAIResponseIntent&#x27;;\n    },\n    async handle(handlerInput) {\n
const { intent } = handlerInput.requestEnvelope.request;\n        \n        // Check
if all slots are filled\n        if (intent.slots.query.value === undefined ||
intent.slots.query.value === &quot;&quot;) {\n            const speakOutput = &quot;I
didn&#x27;t catch that. Can you please ask me a question like, &#x27;What is quantum
mechanics?&#x27;&quot;;\n            return
handlerInput.responseBuilder\n                .speak(speakOutput)\n                .re
prompt(speakOutput)\n                .getResponse();\n        }\n        \n        //
Ensure no delegation\n        handlerInput.requestEnvelope.request.dialogState =
&#x27;COMPLETED&#x27;;\n\n        let aiResponse = &#x27;&#x27;;\n        try {\n
aiResponse = await getAIResponse(intent.slots.query.value);  // Call OpenAI API
here\n        } catch (error) {\n            console.log(`Error with OpenAI API call:
${error}`);\n            aiResponse = &#x27;Sorry, I had trouble processing
that.&#x27;;\n        }\n\n        return
handlerInput.responseBuilder\n            .speak(aiResponse)\n            .getResponse
();\n    }\n};\n```\n\n#### 2. **Dialog State Handling**\nIn Alexa, when dialog
delegation is enabled, Alexa checks the **dialogState** (which could be `STARTED`,
`IN_PROGRESS`, or `COMPLETED`). If it\u2019s not `COMPLETED`, Alexa might assume that
slot values need to be confirmed or filled. By setting the `dialogState` to
`&#x27;COMPLETED&#x27;`, you&#x27;re telling Alexa that everything is ready and it
should process the query without further prompting.\n\n### Recap:\n1. **Ensure No
Dialog Delegation**: By setting the `dialogState` to `COMPLETED`, you tell Alexa that
no further prompts are needed, so it won\u2019t interfere with your skill logic.\n2.
**Handle Missing Slot Values**: Add fallback logic in your handler to check whether
the `query` slot is filled and prompt the user for a question if it&#x27;s missing.\n3.
**Avoid Using Dialog.Directives**: If dialog delegation is not necessary, avoid using
dialog directives in your code. You can handle all slot values and logic directly
within your skill.\n\n### Next Steps:\n- Try adjusting the
`GetAIResponseIntentHandler` to set `dialogState` to `COMPLETED` and ensure all slot
values are handled by your skill (rather than Alexa).\n- Test your skill in the Alexa
Developer Console, making sure it triggers the correct intent.\n\nLet me know if this
resolves the issue or if you need further adjustments!"]}, "status":
"finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "aaa2896d-5aa1-43be-81d9-9d468b149fb9", "finish_details": {"type": "stop",
"stop_tokens": [200002]}, "is_complete": true, "request_id": "8d1b480d4eee4bf3-MXP",
"timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "aaa2896d-
5aa1-43be-81d9-9d468b149fb9", "children": ["aaa2eebc-fbae-4b07-95a4-1a1305adda85"]},
"aaa2eebc-fbae-4b07-95a4-1a1305adda85": {"id": "aaa2eebc-fbae-4b07-95a4-1a1305adda85",
"message": {"id": "aaa2eebc-fbae-4b07-95a4-1a1305adda85", "author": {"role": "user",
"name": null, "metadata": {}}, "create_time": 1728780439.983314, "update_time": null,
"content": {"content_type": "text", "parts": ["so i can replace getAI... with this
code: \n\nconst GetAIResponseIntentHandler = {\r\n    canHandle(handlerInput) {\r\n
return Alexa.getRequestType(handlerInput.requestEnvelope) ===
&#x27;IntentRequest&#x27;\r\n            &amp;&amp;
Alexa.getIntentName(handlerInput.requestEnvelope) ===
```

```
&#x27;GetAIResponseIntent&#x27;;\r\n    },\r\n    async handle(handlerInput) {\r\n
const { intent } = handlerInput.requestEnvelope.request;\r\n        \r\n        //
Check if all slots are filled\r\n        if (intent.slots.query.value === undefined ||
intent.slots.query.value === &quot;&quot;) {\r\n            const speakOutput =
&quot;I didn&#x27;t catch that. Can you please ask me a question like, &#x27;What is
quantum mechanics?&#x27;&quot;;\r\n            return
handlerInput.responseBuilder\r\n                .speak(speakOutput)\r\n
 .reprompt(speakOutput)\r\n                .getResponse();\r\n        }\r\n
\r\n        // Ensure no delegation\r\n
handlerInput.requestEnvelope.request.dialogState = &#x27;COMPLETED&#x27;;\r\n\r\n
let aiResponse = &#x27;&#x27;;\r\n        try {\r\n            aiResponse = await
getAIResponse(intent.slots.query.value);  // Call OpenAI API here\r\n        } catch
(error) {\r\n            console.log(`Error with OpenAI API call: ${error}`);\r\n
aiResponse = &#x27;Sorry, I had trouble processing that.&#x27;;\r\n        }\r\n\r\n
return
handlerInput.responseBuilder\r\n            .speak(aiResponse)\r\n            .getResp
onse();\r\n    }\r\n};\r"]}, "status": "finished_successfully", "end_turn": null,
"weight": 1.0, "metadata": {"serialization_metadata": {"custom_symbol_offsets": []},
"request_id": "8d1b5153af314bf3-MXP", "message_source": null, "timestamp_": "absolute",
"message_type": null}, "recipient": "all", "channel": null}, "parent": "2b2c0aca-135e-
4f90-911d-dce0360bd150", "children": ["f7cfb4eb-c266-4cd2-85dc-aaa389c56a43"]},
"f7cfb4eb-c266-4cd2-85dc-aaa389c56a43": {"id": "f7cfb4eb-c266-4cd2-85dc-aaa389c56a43",
"message": {"id": "f7cfb4eb-c266-4cd2-85dc-aaa389c56a43", "author": {"role":
"assistant", "name": null, "metadata": {}}, "create_time": 1728780446.7264,
"update_time": null, "content": {"content_type": "text", "parts": ["Yes, you can
replace your current `GetAIResponseIntentHandler` code with the updated version
you&#x27;ve shared. This updated handler will ensure the following:\n\n1. **Slot
Handling**: It checks whether the `query` slot has been filled. If the slot is empty
or undefined, Alexa will prompt the user for input (&quot;I didn&#x27;t catch
that...&quot;).\n2. **Disable Dialog Delegation**: The `dialogState` is set to
`COMPLETED` to ensure that Alexa doesn&#x27;t try to handle the dialog automatically
or expect further slot confirmations.\n3. **AI Response Handling**: If the slot is
filled, it will proceed to call the `getAIResponse` function and handle any API errors
gracefully.\n\n### Steps:\n1. **Replace** your existing `GetAIResponseIntentHandler`
code with this updated version.\n2. **Save and Deploy** your skill again.\n3. **Test**
to ensure that Alexa behaves correctly and doesn&#x27;t try to delegate slot filling
or handle the dialog.\n\nThis modification should help ensure that Alexa doesn&#x27;t
interfere with the dialog flow and that your skill processes user queries
correctly.\n\nLet me know how it works out!"]}, "status":
"finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "aaa2eebc-fbae-4b07-95a4-1a1305adda85", "finish_details": {"type": "stop",
"stop_tokens": [200002]}, "is_complete": true, "request_id": "8d1b5153af314bf3-MXP",
"timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "aaa2eebc-
fbae-4b07-95a4-1a1305adda85", "children": ["aaa2100a-265f-4e97-b84f-119fc3cf2d1a"]},
"aaa2100a-265f-4e97-b84f-119fc3cf2d1a": {"id": "aaa2100a-265f-4e97-b84f-119fc3cf2d1a",
"message": {"id": "aaa2100a-265f-4e97-b84f-119fc3cf2d1a", "author": {"role": "user",
"name": null, "metadata": {}}, "create_time": 1728781442.149495, "update_time": null,
"content": {"content_type": "text", "parts": ["this was the json input:
\n\n{\n\t&quot;version&quot;: &quot;1.0&quot;,\n\t&quot;session&quot;:
{\n\t\t&quot;new&quot;: false,\n\t\t&quot;sessionId&quot;: &quot;amzn1.echo-
api.session.db498278-fedf-407f-9bba-0dc209f50a0f&quot;,\n\t\t&quot;application&quot;:
{\n\t\t\t&quot;applicationId&quot;: &quot;amzn1.ask.skill.4e3cd483-a6cc-4494-b190-
11c9b68be961&quot;\n\t\t},\n\t\t&quot;attributes&quot;: {},\n\t\t&quot;user&quot;:
{\n\t\t\t&quot;userId&quot;:
&quot;amzn1.ask.account.AMA5J5IPANGXEWCPW3SHALU2NB4DHXU3BJAYIQQH5UIQAXPOYUACOIEVDTOG22
CEN2SCF7KGF5RJSUCT7ZBLK43TT3DHZFNVUO7KWN2ER26Z234DMXDGSQS4B4A5XMVL6FSWMB7D3BFXS7II5EI2
KOIZDI5FWQRR2YMT4VQHW6ERQTAWO2K3UNS2OJC3M2HN4EIGPSL2P67YCMOL4RZBUDD4RJDZYH5Y2TL4GI7M27
WQ&quot;\n\t\t}\n\t},\n\t&quot;context&quot;: {\n\t\t&quot;Viewports&quot;:
[\n\t\t\t{\n\t\t\t\t&quot;type&quot;: &quot;APL&quot;,\n\t\t\t\t&quot;id&quot;:
&quot;medHub&quot;,\n\t\t\t\t&quot;shape&quot;:
&quot;RECTANGLE&quot;,\n\t\t\t\t&quot;dpi&quot;:
160,\n\t\t\t\t&quot;presentationType&quot;:
&quot;OVERLAY&quot;,\n\t\t\t\t&quot;canRotate&quot;:
false,\n\t\t\t\t&quot;configuration&quot;: {\n\t\t\t\t\t&quot;current&quot;:
{\n\t\t\t\t\t\t&quot;mode&quot;: &quot;HUB&quot;,\n\t\t\t\t\t\t&quot;video&quot;:
{\n\t\t\t\t\t\t\t&quot;codecs&quot;:
[\n\t\t\t\t\t\t\t\t&quot;H_264_42&quot;,\n\t\t\t\t\t\t\t\t&quot;H_264_41&quot;\n\t\t\t
\t\t\t\t]\n\t\t\t\t\t\t},\n\t\t\t\t\t\t&quot;size&quot;:
{\n\t\t\t\t\t\t\t&quot;type&quot;:
&quot;DISCRETE&quot;,\n\t\t\t\t\t\t\t&quot;pixelWidth&quot;:
1280,\n\t\t\t\t\t\t\t&quot;pixelHeight&quot;:
800\n\t\t\t\t\t\t}\n\t\t\t\t\t}\n\t\t\t\t}\n\t\t\t}\n\t\t],\n\t\t&quot;Viewport&quot;:
{\n\t\t\t&quot;experiences&quot;: [\n\t\t\t\t{\n\t\t\t\t\t&quot;arcMinuteWidth&quot;:
221,\n\t\t\t\t\t&quot;arcMinuteHeight&quot;: 162,\n\t\t\t\t\t&quot;canRotate&quot;:
false,\n\t\t\t\t\t&quot;canResize&quot;:
false\n\t\t\t\t}\n\t\t\t],\n\t\t\t&quot;mode&quot;:
&quot;HUB&quot;,\n\t\t\t&quot;shape&quot;:
&quot;RECTANGLE&quot;,\n\t\t\t&quot;pixelWidth&quot;:
1280,\n\t\t\t&quot;pixelHeight&quot;: 800,\n\t\t\t&quot;dpi&quot;:
160,\n\t\t\t&quot;currentPixelWidth&quot;: 1280,\n\t\t\t&quot;currentPixelHeight&quot;:
800,\n\t\t\t&quot;touch&quot;:
[\n\t\t\t\t&quot;SINGLE&quot;\n\t\t\t],\n\t\t\t&quot;keyboard&quot;:
[\n\t\t\t\t&quot;DIRECTION&quot;\n\t\t\t],\n\t\t\t&quot;video&quot;:
{\n\t\t\t\t&quot;codecs&quot;:
[\n\t\t\t\t\t&quot;H_264_42&quot;,\n\t\t\t\t\t&quot;H_264_41&quot;\n\t\t\t\t]\n\t\t\t}
\n\t\t},\n\t\t&quot;Extensions&quot;: {\n\t\t\t&quot;available&quot;:
{\n\t\t\t\t&quot;aplext:backstack:10&quot;:
{}\n\t\t\t}\n\t\t},\n\t\t&quot;System&quot;: {\n\t\t\t&quot;application&quot;:
{\n\t\t\t\t&quot;applicationId&quot;: &quot;amzn1.ask.skill.4e3cd483-a6cc-4494-b190-
11c9b68be961&quot;\n\t\t\t},\n\t\t\t&quot;user&quot;: {\n\t\t\t\t&quot;userId&quot;:
&quot;amzn1.ask.account.AMA5J5IPANGXEWCPW3SHALU2NB4DHXU3BJAYIQQH5UIQAXPOYUACOIEVDTOG22
CEN2SCF7KGF5RJSUCT7ZBLK43TT3DHZFNVUO7KWN2ER26Z234DMXDGSQS4B4A5XMVL6FSWMB7D3BFXS7II5EI2
KOIZDI5FWQRR2YMT4VQHW6ERQTAWO2K3UNS2OJC3M2HN4EIGPSL2P67YCMOL4RZBUDD4RJDZYH5Y2TL4GI7M27
WQ&quot;\n\t\t\t},\n\t\t\t&quot;device&quot;: {\n\t\t\t\t&quot;deviceId&quot;:
&quot;amzn1.ask.device.AMAXRWYUCF3SDX4KFOK4JR3J2LBF4R5I5CM7OLTUQFW2JXKXD6RWEGOMWI6RG5U
XMITSMS4UY6JVTA2OYWS7QAC7TGT5OUMHLI4MHFWQAHZJKLGHZJHCQ3GROHVFZ5K5GG2FQIAI7UHBTO6PYNAU2
```

SE6CYHVUWGDS43ZPD2QON5IX52F545Q3W52JY5LFD43TP5K2WNYDFCAT72T&quot;,\n\t\t\t\t&quot;supp
ortedInterfaces&quot;: {}\n\t\t\t},\n\t\t\t&quot;apiEndpoint&quot;:
&quot;https://api.eu.amazonalexa.com&quot;,\n\t\t\t&quot;apiAccessToken&quot;:
&quot;eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImtpZCI6IjEifQ.eyJhdWQiOiJodHRwczovL2FwaS5ld
S5hbWF6b25hbGV4YS5jb20iLCJpc3MiOiJBbGV4YVNraWxsS2l0Iiwic3ViIjoiYW16bjEuYXNrLnNraWxsLjR
lM2NkNDgzLWE2Y2MtNDQ5NC1iMTkwLTExYzliNjhiZTk2MSIsImV4cCI6MTcyODc4MTEyMiwiaWF0IjoxNzI4N
zgxMDYyLCJuYmYiOjE3Mjg3ODEwNjIsInByaXZhdGVDbGFpbXMiOnsiaXNGb3JSQSI6ImZhbHNlIiwiY29udGV
4dCI6IkFBQUFBQUFBQVFCSWNCbG5lYWJGOVUxSzd3VjRxeW1GZUFFQUFBQUFBQUFnSXJKUm5LZ1JnYjA3c0pSd
TA2YkxEcTRXWGlVUXdJNWc2dFhvNUloZUR2M0JucnBtWUw4cTRhS05hajBnMmpSbktjdGdHeC9pRCswekRGWHE
yWWVCUG44T0NYeFJWNjlwdUpySDh6WmlMbG9xZWMwRW0wWmdTSGwzSjJhWEdsTC9rc3F0aDlMNG1QVUt0a2pIR
HU5SE83dFlzalNVcGOwNk1yVGowZ21ERlM2MXhYMFV0NWNwbDJTOFVBcnFZOXJncE9zb2RSdnNlaldDZ2FxNG5
vVOZXdVdGWThoU2dISWR5anpjZnBxUHJ1eU1FR21nNjI3VnJsNldMbGdkRlZhb2hnVWkrWlZUTytBRTQ4OHB4c
VhLLzc4dE9MU3lHaHlOckhOQkk1ekU3Z3FWWkFYaWdrRDRaSHZRdkJ2cU1ES2NYbmhYdXhoT3JNVkFqdk9KZGF
yQXlnalNZb3B4ejZweWwya09OQ3U0dzV1L2NEaUFzSUo5d0dLM0trVkplcTJxTXNJdytVRzBFVjRuVW9VZHpoS
HdRM3NuaEhsV0RnRXJHcmRteWRhd0wvSHhNTXR2cllzcm9rWUwwR0ZWYm1aQXhZUkVPUlRicTJHdTBhRUZPWGE
0VVc0QWJJWG1kdGRqVUFKZUZuTk1ZUUI5K0dHREh5WkUiLCJkZXZpY2VJZCI6ImFtem4xLmFzay5kZXZpY2UuQ
U1BWFJXWVVDRjNTRFg0S0ZPSzRKUjNKMkxCRjRSNUk1Q003T0xUVVFGVzJKWEtYRDZSV0VHT01XSTZSRzVVWE1
JVFNNUzRVWTZKVlRBMk9ZV1M3UUFDN1RHVDVPVU1ITEk0TUhGV1FBSFpKS0xHSFpKSENRM0dST0hWRlo1SzVHR
zJGUUlBSTdVSEJUTzZQWU5BVTJTRTZDWUhWVVdHRFM0M1pQRDJRT041SVg1MkY1NDVRM1c1MkpZNUxGRDQzVFA
1SzJXTllERkNBVDcyVCIsInVzZXJJZCI6ImFtem4xLmFzay5hY2NvdW50LkFNQTVKNUlQQU5HWEVXQ1BXM1NIQ
UxVMk5CNERIWFUzQkpBWUlRUUg1VUlRQVhQT1lVQUNPSUVWRFRPRzIyQ0VOM1NDRjdLR0Y1UkpTVUNUN1pCTEs
OM1RUMORIWkZOVlVPN0tXTjJFUjI2WjIzNERNWERHU1FTNEI0QTVYTVZMNkZTV01CN0QzQkZYUzdJSTVFSTJLT
OlaREk1RldRUlIyWU1UNFZRSFc2RVJRVEFXTzJLM1VOUzJPSkMzTTJITjRFSUdQUOwyUDY3WUNNTOwOUlpCVUR
ENFJKRFpZSDVZMlRMNEdJNOOyN1dRIn19.art3jk8DJ_Bzi6A6yjk3wMJ2E3OzFsqFhJFQYtsNuGCcG67QhSqT
ou8Sucs1ptvh4zDrD9rcMqlSa0-mk-9tKKjxQpkWbC9T8NK0qGruHBmwuNEt0YminLu-QgbSwRGhKtt-
EdqMESwc-mI8dRf4krdPnwaL5CL5kod6A4eWiaLYr8EFJxjmctyyXlN_z2i-
UP5f61QbrCgg3DwzvAJXAddtySTSpJdAPsjb8IlD7g2ilnZ3IPEiW3Aovt7s2NrLNgeDt1DfN6atsXyKLJFtZd
KFqWmt0pX7vYW6zX0b01XbxXDrMTmf3NLtGTS4I5xTx8VgufuqkyDQ6tnZkim5EA&quot;\n\t\t}\n\t},\n\
t&quot;request&quot;: {\n\t\t&quot;type&quot;:
&quot;SessionEndedRequest&quot;,\n\t\t&quot;requestId&quot;: &quot;amzn1.echo-
api.request.fbdb86ba-fa71-48ea-b526-41ec792301b0&quot;,\n\t\t&quot;timestamp&quot;:
&quot;2024-10-13T00:57:43Z&quot;,\n\t\t&quot;locale&quot;: &quot;en-
US&quot;,\n\t\t&quot;reason&quot;: &quot;ERROR&quot;,\n\t\t&quot;error&quot;:
{\n\t\t\t&quot;type&quot;: &quot;INVALID_RESPONSE&quot;,\n\t\t\t&quot;message&quot;:
&quot;An exception occurred while dispatching the request to the
skill.&quot;\n\t\t}\n\t}\n}\n\nThis was the output :\n\n{\n\t&quot;body&quot;:
{\n\t\t&quot;version&quot;: &quot;1.0&quot;,\n\t\t&quot;response&quot;:
{\n\t\t\t&quot;type&quot;:
&quot;_DEFAULT_RESPONSE&quot;\n\t\t},\n\t\t&quot;sessionAttributes&quot;:
{},\n\t\t&quot;userAgent&quot;: &quot;ask-node/2.14.0 Node/v16.20.2 sample/hello-
world/v1.2&quot;\n\t}\n}\n\nThe first output below returned nothing
\n\n{\n\t&quot;body&quot;: {\n\t\t&quot;version&quot;:
&quot;1.0&quot;,\n\t\t&quot;response&quot;: {\n\t\t\t&quot;type&quot;:
&quot;_DEFAULT_RESPONSE&quot;\n\t\t},\n\t\t&quot;sessionAttributes&quot;:
{},\n\t\t&quot;userAgent&quot;: &quot;ask-node/2.14.0 Node/v16.20.2 sample/hello-
world/v1.2&quot;\n\t}\n}\n\nand overal I recieved the error of: There was a problem
with the requested skill&#x27;s response for the prompt/phrase of &quot;tell me about
the intersection between cops and donuts&quot; what you think might be goin on Chat G?
"]}, "status": "finished_successfully", "end_turn": null, "weight": 1.0, "metadata":

{"serialization_metadata": {"custom_symbol_offsets": []}, "request_id": "8d1b69ca2bb4ba89-MXP", "message_source": null, "timestamp_": "absolute", "message_type": null}, "recipient": "all", "channel": null}, "parent": "f7cfb4eb-c266-4cd2-85dc-aaa389c56a43", "children": ["7079d2bd-c2c3-45ed-bbee-b541330265af"]}, "7079d2bd-c2c3-45ed-bbee-b541330265af": {"id": "7079d2bd-c2c3-45ed-bbee-b541330265af", "message": {"id": "7079d2bd-c2c3-45ed-bbee-b541330265af", "author": {"role": "assistant", "name": null, "metadata": {}}, "create_time": 1728781482.955342, "update_time": null, "content": {"content_type": "text", "parts": ["It looks like the **SessionEndedRequest** in the JSON payload indicates an error occurred during the dispatch of the request, and the error message is of type **INVALID_RESPONSE**, which means there was an issue with how your skill handled the response. Here are some possible reasons and ways to troubleshoot:\n\n### Common Causes of &quot;INVALID_RESPONSE&quot; Errors:\n1. **OpenAI API Error**:\n   - If your OpenAI API request failed or returned an invalid response, your Alexa skill could crash, which would lead to the `INVALID_RESPONSE` error. This might happen if the OpenAI response is malformed or if there&#x27;s an issue with your API request logic.\n\n2. **Response Format Issue**:\n   - Alexa expects a valid response structure, and if your code didn\u2019t return the correct response format or returned an empty response, Alexa would flag it as `INVALID_RESPONSE`.\n\n3. **Timeout from API Call**:\n   - If the OpenAI API call takes too long (beyond the time Alexa allows for skill responses), this might result in a timeout, leading to this type of error.\n\n### Steps to Troubleshoot and Fix:\n\n#### 1. **Check the OpenAI API Call**\n   - Ensure that the OpenAI API is returning a valid response. You can add some logging to verify the API call and response:\n\n```javascript\nasync function getAIResponse(query) {\n   const apiKey = &#x27;your-openai-api-key&#x27;; // Replace with your key\n   const options = {\n   hostname: &#x27;api.openai.com&#x27;,\n   path: &#x27;/v1/completions&#x27;,\n   method: &#x27;POST&#x27;,\n   headers: {\n   &#x27;Authorization&#x27;: `Bearer ${apiKey}`,\n   &#x27;Content-Type&#x27;: &#x27;application/json&#x27;\n   }\n   };\n\n   const body = JSON.stringify({\n   model: &#x27;text-davinci-003&#x27;, // Adjust if needed for other models\n   prompt: query,\n   max_tokens: 100,\n   temperature: 0.7\n   });\n\n   return new Promise((resolve, reject) =&gt; {\n   const req = https.request(options, (res) =&gt; {\n   let data = &#x27;&#x27;;\n   res.on(&#x27;data&#x27;, (chunk) =&gt; {\n   data += chunk;\n   });\n   res.on(&#x27;end&#x27;, () =&gt; {\n   console.log(&#x27;OpenAI API response:&#x27;, data); // Log the raw response\n   const response = JSON.parse(data);\n   resolve(response.choices[0].text.trim());\n   });\n   });\n\n   req.on(&#x27;error&#x27;, (e) =&gt; {\n   console.log(&#x27;OpenAI API call error:&#x27;, e); // Log the error\n   reject(e);\n   });\n\n   req.write(body);\n   req.end();\n   });\n}\n```\n\n- This will log the raw response from OpenAI to check if it&#x27;s properly structured. If the API call returns something unexpected, it could crash the skill.\n- You can also handle **network errors** or **API rate limits** better by wrapping the API call in more extensive error handling.\n\n#### 2. **Return a Valid Alexa Response**\nMake sure that your skill always returns a valid response back to Alexa, even if the OpenAI API call fails. Update your `GetAIResponseIntentHandler` to handle cases where the API fails or times out:\n\n```javascript\nconst GetAIResponseIntentHandler = {\n   canHandle(handlerInput) {\n   return

```
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;GetAIResponseIntent&#x27;;\n    },\n    async handle(handlerInput) {\n
const { intent } = handlerInput.requestEnvelope.request;\n        \n        // Check
if the slot is filled\n        if (intent.slots.query.value === undefined ||
intent.slots.query.value === &quot;&quot;) {\n            const speakOutput = &quot;I
didn&#x27;t catch that. Can you please ask me a question like, &#x27;What is quantum
mechanics?&#x27;&quot;;\n            return
handlerInput.responseBuilder\n                .speak(speakOutput)\n                .re
prompt(speakOutput)\n                .getResponse();\n        }\n\n        let
aiResponse = &#x27;&#x27;;\n        try {\n            aiResponse = await
getAIResponse(intent.slots.query.value);  // Call OpenAI API\n
console.log(&#x27;AI response:&#x27;, aiResponse);  // Log AI response for
debugging\n        } catch (error) {\n            console.log(&#x27;Error
occurred:&#x27;, error);  // Log error for debugging\n            aiResponse =
&#x27;Sorry, I had trouble processing your request.&#x27;;\n        }\n\n        //
Return the AI response or error message\n        return
handlerInput.responseBuilder\n            .speak(aiResponse)\n            .getResponse
();\n    }\n};\n```\n\n- **Log OpenAI Responses**: You can use `console.log` to debug
and track the response from OpenAI.\n- **Fallback Message**: If the API call fails or
takes too long, ensure Alexa returns a fallback message instead of crashing.\n\n#### 3.
**Check for Timeouts**\n   - Alexa has a limited response time (typically around 8
seconds). If the API call takes too long, it could result in an `INVALID_RESPONSE`. To
troubleshoot this, add a timeout or optimize the API call. If OpenAI\u2019s response
is slow, you might want to use shorter prompts or reduce the `max_tokens`.\n\n#### 4.
**Test Locally**\n   - Use the **Alexa Developer Console\u2019s Test tab** to send
specific requests and see what responses are returned. Check the logs to ensure the
API call works correctly and that your skill is processing the response
properly.\n\n#### 5. **Mock OpenAI Response for Testing**\n   - Temporarily mock the
OpenAI API response to test the skill without waiting for a real API
call:\n\n```javascript\nasync function getAIResponse(query) {\n    return new
Promise((resolve) =&gt; {\n        resolve(&quot;This is a mock response from the
AI.&quot;);\n    });\n}\n```\n\n- This will allow you to test the rest of your skill
logic and ensure Alexa\u2019s response format is correct.\n\n### Recap of What to
Do:\n1. **Log the OpenAI API response** to debug whether the API is returning a valid
response.\n2. **Ensure your skill always returns a valid Alexa response**, even if the
API call fails.\n3. **Handle timeouts** by checking if the API response takes too long
and providing fallback responses if needed.\n4. **Test in the Alexa Developer
Console** and analyze the logs for more detailed error messages.\n\nBy following these
steps, you can identify the issue and fix the `INVALID_RESPONSE` error. Let me know
how it goes or if you need more help!"]}, "status": "finished_successfully","end_turn":
true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "aaa2100a-265f-4e97-b84f-119fc3cf2d1a", "finish_details": {"type": "stop",
"stop_tokens": [200002]}, "is_complete": true, "request_id": "8d1b69ca2bb4ba89-MXP",
"timestamp_": "absolute"}, "recipient": "all", "channel": null}, "parent": "aaa2100a-
265f-4e97-b84f-119fc3cf2d1a", "children": ["aaa2ae60-4893-462b-a724-26d470ef6acc"]},
"aaa2ae60-4893-462b-a724-26d470ef6acc": {"id": "aaa2ae60-4893-462b-a724-26d470ef6acc",
```

```
"message": {"id": "aaa2ae60-4893-462b-a724-26d470ef6acc", "author": {"role": "user",
"name": null, "metadata": {}}, "create_time": 1728781968.001932, "update_time": null,
"content": {"content_type": "text", "parts": ["This is JSon  input 1 which recieved no
response: \n\n{\n\t&quot;version&quot;: &quot;1.0&quot;,\n\t&quot;session&quot;:
{\n\t\t&quot;new&quot;: true,\n\t\t&quot;sessionId&quot;: &quot;amzn1.echo-
api.session.747d9edc-e979-4ebf-8926-e8ca2b5820a3&quot;,\n\t\t&quot;application&quot;:
{\n\t\t\t&quot;applicationId&quot;: &quot;amzn1.ask.skill.4e3cd483-a6cc-4494-b190-
11c9b68be961&quot;\n\t\t},\n\t\t&quot;attributes&quot;: {},\n\t\t&quot;user&quot;:
{\n\t\t\t&quot;userId&quot;:
&quot;amzn1.ask.account.AMA5J5IPANGXEWCPW3SHALU2NB4DHXU3BJAYIQQH5UIQAXPOYUACOIEVDTOG22
CEN2SCF7KGF5RJSUCT7ZBLK43TT3DHZFNVUO7KWN2ER26Z234DMXDGSQS4B4A5XMVL6FSWMB7D3BFXS7II5EI2
KOIZDI5FWQRR2YMT4VQHW6ERQTAWO2K3UNS2OJC3M2HN4EIGPSL2P67YCMOL4RZBUDD4RJDZYH5Y2TL4GI7M27
WQ&quot;\n\t\t},\n\t\t&quot;affiliatedResources&quot;: []\n\t},\n\t&quot;context&quot;:
{\n\t\t&quot;Viewports&quot;: [\n\t\t\t{\n\t\t\t\t&quot;type&quot;:
&quot;APL&quot;,\n\t\t\t\t&quot;id&quot;:
&quot;medHub&quot;,\n\t\t\t\t&quot;shape&quot;:
&quot;RECTANGLE&quot;,\n\t\t\t\t&quot;dpi&quot;:
160,\n\t\t\t\t&quot;presentationType&quot;:
&quot;OVERLAY&quot;,\n\t\t\t\t&quot;canRotate&quot;:
false,\n\t\t\t\t&quot;configuration&quot;: {\n\t\t\t\t\t&quot;current&quot;:
{\n\t\t\t\t\t\t&quot;mode&quot;: &quot;HUB&quot;,\n\t\t\t\t\t\t&quot;video&quot;:
{\n\t\t\t\t\t\t\t&quot;codecs&quot;:
[\n\t\t\t\t\t\t\t\t&quot;H_264_42&quot;,\n\t\t\t\t\t\t\t\t&quot;H_264_41&quot;\n\t\t\t
\t\t\t\t]\n\t\t\t\t\t\t},\n\t\t\t\t\t\t&quot;size&quot;:
{\n\t\t\t\t\t\t\t&quot;type&quot;:
&quot;DISCRETE&quot;,\n\t\t\t\t\t\t\t&quot;pixelWidth&quot;:
1280,\n\t\t\t\t\t\t\t&quot;pixelHeight&quot;:
800\n\t\t\t\t\t\t}\n\t\t\t\t\t}\n\t\t\t\t}\n\t\t\t}\n\t\t],\n\t\t&quot;Viewport&quot;:
{\n\t\t\t&quot;experiences&quot;: [\n\t\t\t\t{\n\t\t\t\t\t&quot;arcMinuteWidth&quot;:
221,\n\t\t\t\t\t&quot;arcMinuteHeight&quot;: 162,\n\t\t\t\t\t&quot;canRotate&quot;:
false,\n\t\t\t\t\t&quot;canResize&quot;:
false\n\t\t\t\t}\n\t\t\t],\n\t\t\t&quot;mode&quot;:
&quot;HUB&quot;,\n\t\t\t&quot;shape&quot;:
&quot;RECTANGLE&quot;,\n\t\t\t&quot;pixelWidth&quot;:
1280,\n\t\t\t&quot;pixelHeight&quot;: 800,\n\t\t\t&quot;dpi&quot;:
160,\n\t\t\t&quot;currentPixelWidth&quot;: 1280,\n\t\t\t&quot;currentPixelHeight&quot;:
800,\n\t\t\t&quot;touch&quot;:
[\n\t\t\t\t&quot;SINGLE&quot;\n\t\t\t],\n\t\t\t&quot;keyboard&quot;:
[\n\t\t\t\t&quot;DIRECTION&quot;\n\t\t\t],\n\t\t\t&quot;video&quot;:
{\n\t\t\t\t&quot;codecs&quot;:
[\n\t\t\t\t\t&quot;H_264_42&quot;,\n\t\t\t\t\t&quot;H_264_41&quot;\n\t\t\t\t]\n\t\t\t}
\n\t\t},\n\t\t&quot;Extensions&quot;: {\n\t\t\t&quot;available&quot;:
{\n\t\t\t\t&quot;aplext:backstack:10&quot;:
{}\n\t\t\t}\n\t\t},\n\t\t&quot;Advertising&quot;: {\n\t\t\t&quot;advertisingId&quot;:
&quot;00000000-0000-0000-0000-000000000000&quot;,\n\t\t\t&quot;limitAdTracking&quot;:
true\n\t\t},\n\t\t&quot;System&quot;: {\n\t\t\t&quot;application&quot;:
{\n\t\t\t\t&quot;applicationId&quot;: &quot;amzn1.ask.skill.4e3cd483-a6cc-4494-b190-
11c9b68be961&quot;\n\t\t\t},\n\t\t\t&quot;user&quot;: {\n\t\t\t\t&quot;userId&quot;:
&quot;amzn1.ask.account.AMA5J5IPANGXEWCPW3SHALU2NB4DHXU3BJAYIQQH5UIQAXPOYUACOIEVDTOG22
CEN2SCF7KGF5RJSUCT7ZBLK43TT3DHZFNVUO7KWN2ER26Z234DMXDGSQS4B4A5XMVL6FSWMB7D3BFXS7II5EI2
KOIZDI5FWQRR2YMT4VQHW6ERQTAWO2K3UNS2OJC3M2HN4EIGPSL2P67YCMOL4RZBUDD4RJDZYH5Y2TL4GI7M27
WQ&quot;\n\t\t\t},\n\t\t\t&quot;device&quot;: {\n\t\t\t\t&quot;deviceId&quot;:
&quot;amzn1.ask.device.AMAXRWYUCF3SDX4KFOK4JR3J2LBF4R5I5CM7OLTUQFW2JXKXD6RWEGOMWI6RG5U
XMITSMS4UY6JVTA2OYWS7QAC7TGT5OUMHLI4MHFWQAHZJKLGHZJHCQ3GROHVFZ5K5GG2FQIAI7UHBTO6PYNAU2
SE6CYHVUWGDS43ZPD2QON5IX52F545Q3W52JY5LFD43TP5K2WNYDFCAT72T&quot;,\n\t\t\t\t&quot;supp
ortedInterfaces&quot;: {}\n\t\t\t},\n\t\t\t&quot;apiEndpoint&quot;:
&quot;https://api.eu.amazonalexa.com&quot;,\n\t\t\t&quot;apiAccessToken&quot;:
&quot;eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImtpZCI6IjEifQ.eyJhdWQiOiJodHRwczovL2FwaS5ld
S5hbWF6b25hbGV4YS5jb20iLCJpc3MiOiJBbGV4YVNraWxsS2lOIiwic3ViIjoiYW16bjEuYXNrLnNraWxsLjR
lM2NkNDgzLWE2Y2MtNDQ5NC1iMTkwLTExYzliNjhiZTk2MSIsImV4cCI6MTcyODc4MTk0NywiaWF0IjoxNzI4N
zgxODg3LCJuYmYiOjE3Mjg3ODE4ODcsInByaXZhdGVDbGFpbXMiOnsiaXNGb3JSQSI6ImZhbHNlIiwiY29udGV
4dCI6IkFBQUFBQUFBQVFCa2d4TXo2azZGVy9ZWVE0a3JUMW11ZUFFQUFBQUFBQUFJc2M3TXRrb2R2OHhNaW5RV
GdDL3FDN0JhM0tNa0FKMFgrWG5GUElGVE1rMk5MbWJnTGxkR29aYVhENjEyUDhuNXJUNU02VOOxUm9rZ1JlZjh
TcFg5eE1Qclg5VHlUalVCQmN1NGtUaEp5amYrVDVNdk8wMHBkbC9OK3V4Q081M3gxRTZ6azRmbkIOSm9qcUljU
2llWktQMUs4aStQMzBQZ1JMRkFhay84bW9lbHA4VERXUGwrOXRyWWZONGUvVW5vcGtOVyt1Yk15bHhNK1Y1bGd
FSFhaeURaeGtMcm5lL2tCMjdHenNENXBGTkxnV1VDVOxuaklRN3J3V3d1bFM1Z3FocXFOWXJLQOhOQ2R6dOdTU
VgvYTBWam1KNm1KUndMOUJNa2RjNEFFWEhRY3FONFFZVk5HQzdvNHdXVWhzaXUreWM4SVdTaDFVNlFuSlA1WHo
vdORiK1U4THFGUVhsVXlxcVRQd1RmZVlPQUFIL3dOQXVuZGd3YXV4bOdyOS96MWhNVXNxVUwxdVROSOl2YW9rQ
XNBd2JSOXJyWWExUGdqeldJM3ljZGxBa3BXQkc3eU9NWG1RUzFEVO1BMkY2WkMOZ3paeVhkNDZLdmlVMW5QQXF
3RUxEU2tNdWVwUi9pelY4YjdkajFiZWOzOWxpMENzR2wiLCJkZXZpY2VJZCI6ImFtem4xLmFzay5kZXZpY2UuQ
U1BWFJXWVVDRjNTRFg0S0ZPSzRKUjNKMkxCRjRSNUk1Q003T0xUVVFGVzJKWEtYRDZSV0VHT01XSTZSRzVVWE1
JVFNNUzRVWTZKV1RBMk9ZV1M3UUFDN1RHVDVPVU1ITEk0TUhGV1FBSFpKSOxHSFpKSENRM0dST0hWRlo1SzVHR
zJGUU1BSTdVSEJUTzZQWU5BVTJTRTZDWUhWVVdHRFM0M1pQRDJRT041SVg1MkY1NDVRM1c1MkpZNUxGRDQzVFA
1SzJXTllERkNBVDcyVCIsInVzZXJJZCI6ImFtem4xLmFzay5hY2NvdW50LkFNQTVKNUlQQU5HWEVXQ1BXM1NIQ
UxVMk5CNERIWFUzQkpBWU1RUUg1VU1RQVhQT1lVQUNPSUVWRFRPRzIyQOVOM1NDRjdLR0Y1UkpTVUNUN1pCTEs
OM1RUMORIWkZOVlVPNOtXTjJFUjI2WjIzNERNWERHU1FTNEI0QTVYTVZMNkZTV01CN0QzQkZYUzdJSTVFSTJLT
OlaREk1RldRUlIyWU1UNFZRSFc2RVJRVEFXTzJLM1VOUzJPSkMzTTJITjRFSUdQU0wyUDY3WUNNT0w0UlpCVUR
ENFJKRFpZSDVZMlRMNEdJNOOyN1dRIn19.LHJSxYzP5ynfzGznxVOzOQ_7ZOiDLtb1JNkWAXkz7jbEpggVOoU_
GRLBTFxbphyQBLjOyVnMKQOpynAiCh11Nbs4vOCox3lULCLUMmDoOmXUdsa5l66EO__Dq_7ojbjpdaTGqs4UcL
WtKfremOh5RKANi5_pwwSDJ51kwPH7KLLLn43XMRmIDATRP7VxCx28haQFtSmdktxETK5gs9Qd6CBxORTsUelr
OFloePP_aI2kLa3kO-
86DJGQgBBkjwbNSxZ77NrVje4DKdO3HqGOkbHOdWSoYCuOGJUhxQng94qJX_alNOxOzDqsJ_imbv-
kbeWj5XNypKZjsgfv1ytQdQ&quot;\n\t\t}\n\t},\n\t&quot;request&quot;:
{\n\t\t&quot;type&quot;: &quot;LaunchRequest&quot;,\n\t\t&quot;requestId&quot;:
&quot;amzn1.echo-api.request.e081005e-eefd-4d80-944f-
38cdf27b95d0&quot;,\n\t\t&quot;locale&quot;: &quot;en-
US&quot;,\n\t\t&quot;timestamp&quot;: &quot;2024-10-
13T01:11:27Z&quot;,\n\t\t&quot;shouldLinkResultBeReturned&quot;: false\n\t}\n}"]},
"status": "finished_successfully", "end_turn": null, "weight": 1.0, "metadata":
{"serialization_metadata": {"custom_symbol_offsets": []}, "request_id":
"8d1b76a20a1aba89-MXP", "message_source": null, "timestamp_": "absolute",
"message_type": null}, "recipient": "all", "channel": null}, "parent": "7079d2bd-c2c3-
45ed-bbee-b541330265af", "children": ["2d3f3dc6-f1d3-4ebc-a159-c501b3ed8712"]},
"2d3f3dc6-f1d3-4ebc-a159-c501b3ed8712": {"id": "2d3f3dc6-f1d3-4ebc-a159-c501b3ed8712",
"message": {"id": "2d3f3dc6-f1d3-4ebc-a159-c501b3ed8712", "author": {"role":
"assistant", "name": null, "metadata": {}}, "create_time": 1728781988.441769,
"update_time": null, "content": {"content_type": "text", "parts": ["The JSON input
you&#x27;ve shared appears to be a **LaunchRequest** from Alexa, which means the user
```

````
has opened or initiated your skill, but no specific intent has been triggered yet.
This type of request typically comes when a user says something like &quot;Alexa, open
[your skill name].&quot;\n\n### Key Points:\n- **LaunchRequest** doesn&#x27;t carry a
user&#x27;s specific question. It\u2019s just an indication that the user has launched
the skill.\n- After receiving this request, your skill should respond with a greeting
or a prompt that asks the user what they&#x27;d like to do next.\n\n### What Should
Happen Next:\nUpon receiving this **LaunchRequest**, your skill should respond with a
welcome message and prompt the user to ask a question or request something. For
example, it might say, &quot;Welcome! You can ask me about anything. What would you
like to know?&quot;\n\n### Here&#x27;s what the response should look like in your
`LaunchRequestHandler`:\n\n```javascript\nconst LaunchRequestHandler = {\n
canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) ===
&#x27;LaunchRequest&#x27;;\n    },\n    handle(handlerInput) {\n        const
speakOutput = &#x27;Welcome! You can ask me any question, like &quot;Tell me about
artificial intelligence.&quot; What would you like to know?&#x27;;\n        \n
return
handlerInput.responseBuilder\n            .speak(speakOutput)\n            .reprompt(s
peakOutput)  // Adds a reprompt to ensure Alexa keeps
listening\n            .getResponse();\n    }\n};\n```\n\n### What Might Be
Happening:\nIf Alexa receives the **LaunchRequest** and no response is provided, it
could appear like the skill isn\u2019t doing anything. In your skill, if the
`LaunchRequestHandler` is not properly responding, it would lead to no interaction and
might seem like Alexa didn\u2019t respond at all.\n\n### To Test:\n1. **Check the
Logs**: Make sure the `LaunchRequestHandler` is triggering and logging a message.\n2.
**Ensure the Handler is Properly Added**: Make sure the `LaunchRequestHandler` is
added in your skill:\n\n```javascript\nexports.handler =
Alexa.SkillBuilders.custom()\n    .addRequestHandlers(\n        LaunchRequestHandler,
// Ensure this is included\n        HelloWorldIntentHandler,\n
HelpIntentHandler,\n        CancelAndStopIntentHandler,\n
FallbackIntentHandler,\n        GetAIResponseIntentHandler, \n
SessionEndedRequestHandler,\n
IntentReflectorHandler\n    )\n    .addErrorHandlers(ErrorHandler)\n    .lambda();\n``
`\n\n3. **Test Interaction**:\n   - Try saying &quot;Alexa, open [your skill
name]&quot; and ensure it returns a welcome message.\n   - Then, after the
LaunchRequest, test a full query, such as &quot;Tell me about artificial
intelligence.&quot;\n\n### Next Steps:\n- Update your `LaunchRequestHandler` to ensure
it returns a welcome message.\n- Test the full interaction after the LaunchRequest to
ensure the skill moves smoothly from launch to handling user queries.\n\nLet me know
how it goes after making these updates!"]}, "status":
"finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "aaa2ae60-4893-462b-a724-26d470ef6acc", "request_id": "8d1b76a20a1aba89-
MXP", "timestamp_": "absolute", "finish_details": {"type": "stop", "stop_tokens":
[200002]}, "is_complete": true}, "recipient": "all", "channel": null}, "parent":
"aaa2ae60-4893-462b-a724-26d470ef6acc", "children": ["aaa23631-57c2-46ea-9e80-
0b2c6480056c"]}, "aaa23631-57c2-46ea-9e80-0b2c6480056c": {"id": "aaa23631-57c2-46ea-
9e80-0b2c6480056c", "message": {"id": "aaa23631-57c2-46ea-9e80-0b2c6480056c", "author":
````

```
{"role": "user", "name": null, "metadata": {}}, "create_time": 1728782053.700337,
"update_time": null, "content": {"content_type": "text", "parts": ["how do i check the
logs in   the developert test module/section of Alexa"]}, "status":
"finished_successfully", "end_turn": null, "weight": 1.0, "metadata":
{"serialization_metadata": {"custom_symbol_offsets": []}, "request_id":
"8d1b78abba6bba89-MXP", "message_source": null, "timestamp_": "absolute",
"message_type": null}, "recipient": "all", "channel": null}, "parent": "2d3f3dc6-f1d3-
4ebc-a159-c501b3ed8712", "children": ["4f85a1ae-3b83-444b-8993-277c50df5f9e"]},
"4f85a1ae-3b83-444b-8993-277c50df5f9e": {"id": "4f85a1ae-3b83-444b-8993-277c50df5f9e",
"message": {"id": "4f85a1ae-3b83-444b-8993-277c50df5f9e", "author": {"role":
"assistant", "name": null, "metadata": {}}, "create_time": 1728782083.819146,
"update_time": null, "content": {"content_type": "text", "parts": ["To check logs for
your Alexa skill in the Alexa Developer Console, follow these steps:\n\n### 1.
**Enable Logging (CloudWatch) for Your Alexa Skill:**\nTo view logs for your Alexa
skill, you\u2019ll need to enable **Amazon CloudWatch** logging. This allows you to
track what happens when users interact with your skill, including input requests,
errors, and custom log messages you add.\n\n#### Steps to Enable CloudWatch
Logging:\n1. **Go to the Alexa Developer Console**:\n   - Open the [Alexa Developer
Console](https://developer.amazon.com/alexa/console/ask).\n2. **Select Your Skill**:
Choose the skill you are working on from the dashboard.\n3. **Open the Build Tab**: In
the skill, navigate to the **Build** tab.\n4. **Go to Permissions**:\n   - On the left
menu, click **Permissions** under **Build**.\n5. **Enable CloudWatch Logs**:\n   - In
the Permissions section, enable **Amazon CloudWatch Logs** for your skill.\n\n   Once
enabled, all requests and responses, along with any custom logs you&#x27;ve added to
your code, will be available in CloudWatch.\n\n### 2. **Check the Logs in Amazon
CloudWatch:**\nOnce you\u2019ve enabled CloudWatch logging, follow these steps to view
the logs:\n\n1. **Open the AWS Management Console**:\n   - Go to the [AWS Management
Console](https://aws.amazon.com/console/).\n   - Sign in with the AWS account that is
linked to your Alexa skill.\n   \n2. **Go to CloudWatch**:\n   - Search for
**CloudWatch** in the services menu and open it.\n\n3. **Navigate to Logs**:\n   - In
the CloudWatch dashboard, select **Logs** from the left-hand side.\n   \n4. **Find
Your Skill\u2019s Log Group**:\n   - Look for the log group that corresponds to your
skill. It should be named something like:\n     ```\n     /aws/lambda/&lt;Your-Skill-
Name&gt;\n     ```\n   - Select this log group to see all the logs for your
skill.\n\n5. **View the Logs**:\n   - Each log event represents a request
(LaunchRequest, IntentRequest, etc.) that was processed by your skill.\n   - Open the
log stream to view details such as the input request, output response, and any logs
you&#x27;ve added in your code (via `console.log`).\n\n### 3. **Check Logs in the
Alexa Developer Console (Simpler Testing)**:\nFor **quick testing** without going into
CloudWatch, you can use the **Test** tab in the Alexa Developer Console:\n\n1. **Go to
the Alexa Developer Console**:\n   - Open your skill in the [Alexa Developer
Console](https://developer.amazon.com/alexa/console/ask).\n   \n2. **Navigate to the
Test Tab**:\n   - Click on the **Test** tab at the top of the screen.\n   - Ensure the
**Skill Testing** is set to **Development** (switch on at the top).\n\n3. **Use the
Simulator**:\n   - You can type or speak requests directly in the **Alexa Simulator**.
Try invoking your skill by typing: `open [your skill name]`.\n   - After you send a
request, the **JSON input** and **JSON output** will be displayed below the
simulator.\n\n4. **Analyze the Output**:\n   - **Input JSON**: Shows the exact request
Alexa sent to your skill.\n   - **Output JSON**: Displays the response that your skill
```

sent back to Alexa.\n\n   If the output is empty or an error occurs, it\u2019s a sign
there may be issues in your code.\n\n### 4. **Add Logs to Your Code** (Optional but
Helpful):\nTo track issues or check the flow of your skill, you can add
`console.log()` statements in your code:\n\nExample:\n```javascript\nconst
LaunchRequestHandler = {\n   canHandle(handlerInput) {\n   return
Alexa.getRequestType(handlerInput.requestEnvelope) ===
&#x27;LaunchRequest&#x27;;\n   },\n   handle(handlerInput) {\n
console.log(&quot;LaunchRequest triggered&quot;); // Add this to log the request\n
const speakOutput = &#x27;Welcome! You can ask me any question. What would you like to
know?&#x27;;\n\n   return
handlerInput.responseBuilder\n   .speak(speakOutput)\n   .reprompt(s
peakOutput)\n   .getResponse();\n   }\n};\n```\n\n- These logs will be
visible in **CloudWatch**.\n\n### Summary:\n- **Enable CloudWatch** logs in the Alexa
Developer Console.\n- **View Logs**: Use AWS CloudWatch to view detailed logs for your
skill.\n- **Test Tab**: Use the Test tab in the Alexa Developer Console for quick
testing and to view input/output.\n- **Add Logs**: Use `console.log()` in your code to
help track the flow and identify any issues.\n\nLet me know if you need further help
with checking your logs or troubleshooting your skill!"]}, "status":
"finished_successfully","end_turn": true
, "weight": 1.0, "metadata": {"citations": [], "content_references": [], "gizmo_id":
null, "message_type": null, "model_slug": "gpt-4o", "default_model_slug": "gpt-4o",
"parent_id": "aaa23631-57c2-46ea-9e80-0b2c6480056c", "request_id": "8d1b78abba6bba89-
MXP", "timestamp_": "absolute", "finish_details": {"type": "stop", "stop_tokens":
[200002]}, "is_complete": true}, "recipient": "all", "channel": null}, "parent":
"aaa23631-57c2-46ea-9e80-0b2c6480056c", "children": ["aaa2c37f-9c6f-4aa9-9f55-
5f5e6c8ee36c"]}, "aaa2c37f-9c6f-4aa9-9f55-5f5e6c8ee36c": {"id": "aaa2c37f-9c6f-4aa9-
9f55-5f5e6c8ee36c", "message": {"id": "aaa2c37f-9c6f-4aa9-9f55-5f5e6c8ee36c", "author":
{"role": "user", "name": null, "metadata": {}}, "create_time": 1728784309.3566,
"update_time": null, "content": {"content_type": "text", "parts": ["error log: \n\n{\n
&quot;header&quot;: {\n   &quot;namespace&quot;: &quot;SkillDebugger&quot;,\n
&quot;name&quot;: &quot;CaptureError&quot;,\n   &quot;messageId&quot;:
&quot;49e48937-b82c-41b9-8e8f-9815610e8b21&quot;\n   },\n   &quot;payload&quot;: {\n
&quot;skillId&quot;: &quot;amzn1.ask.skill.4e3cd483-a6cc-4494-b190-
11c9b68be961&quot;,\n   &quot;timestamp&quot;: &quot;2024-10-
13T01:43:24.924Z&quot;,\n   &quot;dialogRequestId&quot;: &quot;5fd4a745-8e59-
4a31-ae9f-6d2bdcd194f7&quot;,\n   &quot;skillRequestId&quot;: &quot;amzn1.echo-
api.request.c9fb35f8-8a94-4b30-ac06-7977b90400c4&quot;,\n   &quot;code&quot;:
&quot;SKILL_ENDPOINT_ERROR&quot;,\n   &quot;description&quot;: &quot;Skill
execution returned an exception for requestId amzn1.echo-api.request.c9fb35f8-8a94-
4b30-ac06-7977b90400c4&quot;,\n   &quot;debuggingInfo&quot;: {\n
&quot;type&quot;: &quot;SkillExecutionInfo&quot;,\n   &quot;content&quot;:
{\n   &quot;invocationRequest&quot;: {\n
&quot;endpoint&quot;: &quot;arn:aws:lambda:eu-west-1:975050070604:function:4e3cd483-
a6cc-4494-b190-11c9b68be961:Release_0&quot;,\n   &quot;body&quot;:
{\n   &quot;version&quot;: &quot;1.0&quot;,\n
&quot;session&quot;: {\n   &quot;new&quot;: false,\n
&quot;sessionId&quot;: &quot;amzn1.echo-api.session.fa352e08-94b8-457a-a3e8-
e1a99e388e90&quot;,\n   &quot;application&quot;: {\n
&quot;applicationId&quot;: &quot;amzn1.ask.skill.4e3cd483-a6cc-4494-b190-

```
11c9b68be961&quot;\n                          },\n
&quot;attributes&quot;: {},\n                         &quot;user&quot;: {\n
&quot;userId&quot;:
&quot;amzn1.ask.account.AMA5J5IPANGXEWCPW3SHALU2NB4DHXU3BJAYIQQH5UIQAXPOYUACOIEVDTOG22
CEN2SCF7KGF5RJSUCT7ZBLK43TT3DHZFNVUO7KWN2ER26Z234DMXDGSQS4B4A5XMVL6FSWMB7D3BFXS7II5EI2
KOIZDI5FWQRR2YMT4VQHW6ERQTAWO2K3UNS2OJC3M2HN4EIGPSL2P67YCMOL4RZBUDD4RJDZYH5Y2TL4GI7M27
WQ&quot;,\n                         &quot;permissions&quot;: {\n
&quot;scopes&quot;: {\n
&quot;alexa::devices:all:geolocation:read&quot;: {\n
&quot;status&quot;:
&quot;DENIED&quot;\n                                    }\n
             }\n                                  }\n                         }\n
                    },\n                           &quot;context&quot;: {\n
&quot;Viewports&quot;: [\n                                  {\n
&quot;type&quot;: &quot;APL&quot;,\n                                            &quot;id&quot;:
&quot;medHub&quot;,\n                                    &quot;shape&quot;:
&quot;RECTANGLE&quot;,\n                                 &quot;dpi&quot;: 160,\n
&quot;presentationType&quot;: &quot;OVERLAY&quot;,\n
&quot;canRotate&quot;: false,\n
&quot;configuration&quot;: {\n
&quot;current&quot;: {\n                                        &quot;mode&quot;:
&quot;HUB&quot;,\n                                       &quot;video&quot;: {\n
&quot;codecs&quot;: [\n
&quot;H_264_42&quot;,\n
&quot;H_264_41&quot;\n                                        ]\n
                               },\n
&quot;size&quot;: {\n                                             &quot;type&quot;:
&quot;DISCRETE&quot;,\n
&quot;pixelWidth&quot;: 1280,\n
&quot;pixelHeight&quot;:
800\n                                          }\n
      }\n                                      }\n                                }\n
                         ],\n                           &quot;Viewport&quot;: {\n
&quot;experiences&quot;: [\n                                  {\n
&quot;arcMinuteWidth&quot;: 221,\n
&quot;arcMinuteHeight&quot;: 162,\n
&quot;canRotate&quot;: false,\n
&quot;canResize&quot;:
false\n                                    }\n                               ],\n
&quot;mode&quot;: &quot;HUB&quot;,\n                                     &quot;shape&quot;:
&quot;RECTANGLE&quot;,\n                         &quot;pixelWidth&quot;:
1280,\n                          &quot;pixelHeight&quot;: 800,\n
&quot;dpi&quot;: 160,\n                          &quot;currentPixelWidth&quot;:
1280,\n                          &quot;currentPixelHeight&quot;: 800,\n
&quot;touch&quot;: [\n
&quot;SINGLE&quot;\n                            ],\n
&quot;keyboard&quot;: [\n
&quot;DIRECTION&quot;\n                            ],\n
&quot;video&quot;: {\n                                   &quot;codecs&quot;: [\n
&quot;H_264_42&quot;,\n
&quot;H_264_41&quot;\n                                   ]\n
       }\n                              },\n
&quot;Extensions&quot;: {\n                              &quot;available&quot;: {\n
&quot;aplext:backstack:10&quot;:
{}\n                              }\n                              },\n
&quot;System&quot;: {\n                              &quot;application&quot;: {\n
&quot;applicationId&quot;: &quot;amzn1.ask.skill.4e3cd483-a6cc-4494-b190-
11c9b68be961&quot;\n                              },\n
&quot;user&quot;: {\n                              &quot;userId&quot;:
&quot;amzn1.ask.account.AMA5J5IPANGXEWCPW3SHALU2NB4DHXU3BJAYIQQH5UIQAXPOYUACOIEVDTOG22
CEN2SCF7KGF5RJSUCT7ZBLK43TT3DHZFNVUO7KWN2ER26Z234DMXDGSQS4B4A5XMVL6FSWMB7D3BFXS7II5EI2
KOIZDI5FWQRR2YMT4VQHW6ERQTAWO2K3UNS2OJC3M2HN4EIGPSL2P67YCMOL4RZBUDD4RJDZYH5Y2TL4GI7M27
WQ&quot;,\n                              &quot;permissions&quot;: {\n
&quot;scopes&quot;: {\n
&quot;alexa::devices:all:geolocation:read&quot;: {\n
&quot;status&quot;:
&quot;DENIED&quot;\n                              }\n
                    }\n                              }\n
         },\n                              &quot;device&quot;: {\n
&quot;deviceId&quot;:
&quot;amzn1.ask.device.AMAXRWYUCF3SDX4KFOK4JR3J2LBF4R5I5CM7OLTUQFW2JXKXD6RWEGOMWI6RG5U
XMITSMS4UY6JVTA2OYWS7QAC7TGT5OUMHLI4MHFWQAHZJKLGHZJHCQ3GROHVFZ5K5GG2FQIAI7UHBTO6PYNAU2
SE6CYHVUWGDS43ZPD2QON5IX52F545Q3W52JY5LFD43TP5K2WNYDFCAT72T&quot;,\n
&quot;supportedInterfaces&quot;: {}\n                              },\n
&quot;apiEndpoint&quot;: &quot;https://api.eu.amazonalexa.com&quot;,\n
&quot;apiAccessToken&quot;:
&quot;eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImtpZCI6IjEifQ.eyJhdWQiOiJodHRwczovL2FwaS5ld
S5hbWF6b25hbGV4YS5jb20iLCJpc3MiOiJBbGV4YVNraWxsS2l0Iiwic3ViIjoiYW16bjEuYXNrLnNraWxsLjR
lM2NkNDgzLWE2Y2MtNDQ5NC1iMTkwLTExYzliNjhiZTk2MSIsImV4cCI6MTcyODc4Mzg2MywiaWF0IjoxNzI4N
zgzODAzLCJuYmYiOjE3Mjg3ODM4MDMsInByaXZhdGVDbGFpbXMiOnsiaXNGb3JSQSI6ImZhbHNlIiwibm9uTHd
hU2NvcGVzIjoiYWxleGE6OnBlcnNvbl9pZDpyZWFkIiwiY29udGV4dCI6IkFBQUFBQUFBQVFCa2d4TXo2azZGV
y9ZWVE0a3JUMW11ZUFFQUFBQUFBQURUUHlqMDl4WW90SjB5Zmplb3E4b1MwNmNLQVFIb1Yzd1A4RGFvTGF4TXZ
sY2dDeWYwblVlQzBWY3lDZk1EcjM2NVFUQ1UyaGtnMURRNDZjN2g3T250TXQwdXJyTVl1eHhHUno4Rjc1Mmk1Z
TQ0VjZPY2Z6MlNselBvd0hDWXd4UVk3a013dVkzWVA4bmpQd3dRVlUxTEY3K1V3L3greU5icGZKT2pOMVl6MHd
wbko1bzY2SkVXeFZySXV2bCszZHA4ZzNFaVpUeUxYeXROWG5xZEcyaG5RR2RCbFZsUXBrOW1MZzMvcGRvTzZ5b
mo0WkcrWXM2N2Y2U2paL1R0ZjBqaTlTMW5zUVNHNVB1dENyR3VrZ2JIL1VSVzRPNnlRemdvN01OMEJPNlU1UnA
wTVRWQU1wTFYvVDRFNmVtd1FkZVliTEViQUE1Vk9VZE1SUjhyTjJIUk1ObnRiU1U5cEN2VjY4a1pPWDB6M1oyc
mxRZ3ZuVFNBbFBlYTh2ckdLdDJueGtEVU8xQUVuL1BvM2ozbHFjYUNPWGc5d3ZXT2ZzOWpOZEs5NVBoOXRSbVR
tV0RuMUgrWWRydExUam5VendrYjkxTzlQVll0eDhnQVJTMkw1ZDRkN1FZUjZYVVNVY0krZGlxcU94L1U2akEwU
29yOC9vR1MiLCJkZXZpY2VJZCI6ImFtem4xLmFzay5kZXZpY2UuQU1BWFJXWVVDRjNTRFg0S0ZPSzRKUjNKMkx
CRjRSNUk1Q003T0xUVVFGVzJKWEtYRDZSV0VHT01XSTZSRzVVWE1JVFNNUzRVWTZKVlRBMk9ZV1M3UUFDN1RHV
DVPVU1ITEk0TUhGV1FBSFpKS0xHSFpKSENRM0dST0hWRlo1SzVHRzJGUUlBSTdVSEJUTzZQWU5BVTJTRTZDWUh
WVVdHRFM0M1pQRDJRT041SVg1MkY1NDVRM1c1MkpZNUxGRDQzVFA1SzJXTllERkNBVDcyVCIsInVzZXJJZCI6I
mFtem4xLmFzay5hY2NvdW50LkFNQTVKNUlQQU5HWEVXQ1BXM1NIQUxVMk5CNERIWFUzQkpBWUlRUUg1VUlRQVh
QT1lVQUNPSUVWRFRPRzIyQ0VOMlNDRjdLR0Y1UkpTVUNUN1pCTEs0M1RUM0RIWkZOVlVPN0tXTjJFUjI2WjIzN
ERNWERHU1FTNEI0QTVYTVZMNkZTV01CN0QzQkZYUzdJSTVFSTJLT0laREk1RldRUlIyWU1UNFZRSFc2RVJRVEF
XTzJLM1VOUzJPSkMzTTJITjRFSUdQU0wyUDY3WUNNT0w0UlpCVUREN1JKRFpZSDVZMlRMNEdJN00yN1dRIn19.
Tv9mFHlrChGLSrmu2pco6Tc5iA_OokVs3B6TmOdXirgkynNcMolOVKNIgQnd67mMm5JbAJS69b5jW5MdxCcSN2
```

```
pj42vjLRO4qO-
5tiRnvM7BbdxlHlqXjKlvkVcieHh713t8t64ie9jia6_HvHw6mQ9iSACVSAp338cHylJgDWY1Ooxg7U2EMNirT
wy8WRYtNeNXP3oPgVtabUNhig-
YyhbRw_JYO6n3QBRm9ke8jsXYQnDCueNqF2E6AOJhT8N5tOx2f8UxppxO9dHRANy2CJOAm9Y85LmXfPH-
sTlRAtBZ-
Yor2sqvMjTsGnOuwizZjKHnsUNi_QfObZQ7WIIOZw&quot;\n                    }\n
                 },\n                        &quot;request&quot;: {\n
&quot;type&quot;: &quot;SessionEndedRequest&quot;,\n
&quot;requestId&quot;: &quot;amzn1.echo-api.request.c9fb35f8-8a94-4b30-ac06-
7977b90400c4&quot;,\n                        &quot;timestamp&quot;: &quot;2024-10-
13T01:43:24Z&quot;,\n                        &quot;locale&quot;: &quot;en-
US&quot;,\n                        &quot;reason&quot;: &quot;ERROR&quot;,\n
&quot;error&quot;: {\n                            &quot;type&quot;:
&quot;INVALID_RESPONSE&quot;,\n                            &quot;message&quot;:
&quot;An exception occurred while dispatching the request to the
skill.&quot;\n                        }\n                    }\n
      }\n                },\n                &quot;metrics&quot;: {\n
&quot;skillExecutionTimeInMilliseconds&quot;:
401\n                }\n            }\n        }\n    }\n}"]}, "status":
"finished_successfully", "end_turn": null, "weight": 1.0, "metadata":
{"serialization_metadata": {"custom_symbol_offsets": []}, "request_id":
"8d1bafc70d8f4c43-MXP", "message_source": null, "timestamp_": "absolute",
"message_type": null}, "recipient": "all", "channel": null}, "parent": "4f85a1ae-3b83-
444b-8993-277c50df5f9e", "children": ["bf426d95-2252-45be-a9c6-500ad8d6c415"]},
"bf426d95-2252-45be-a9c6-500ad8d6c415": {"id": "bf426d95-2252-45be-a9c6-500ad8d6c415",
"message": {"id": "bf426d95-2252-45be-a9c6-500ad8d6c415", "author": {"role":
"assistant", "name": null, "metadata": {}}, "create_time": 1728784346.563108,
"update_time": null, "content": {"content_type": "text", "parts": ["The error
you&#x27;re encountering (`INVALID_RESPONSE`) is caused by an issue with how your
skill is processing and returning responses to Alexa. Specifically, Alexa is expecting
a valid response, but an exception occurred, which caused Alexa to terminate the
session and return a **SessionEndedRequest** with the `INVALID_RESPONSE` error.\n\n###
Key Points:\n- **Reason**: The error log indicates that an &quot;exception occurred
while dispatching the request to the skill.&quot;\n- **Root Cause**: This typically
happens if there&#x27;s an error in your code, such as:\n  - An issue with the API
call (e.g., OpenAI API response).\n  - An error in how the response is being
constructed and sent back to Alexa.\n  - A problem with your Lambda function or a
timeout.\n\n### Steps to Fix:\n\n#### 1. **Add Detailed Logging to Debug the
Error**\nTo pinpoint the exact issue, you should add more logging in your handlers,
especially around the areas where exceptions might occur, such as your API call or
response building.\n\nUpdate your `GetAIResponseIntentHandler` as
follows:\n\n```javascript\nconst GetAIResponseIntentHandler = {\n
canHandle(handlerInput) {\n        return
Alexa.getRequestType(handlerInput.requestEnvelope) === &#x27;IntentRequest&#x27;\n
&amp;&amp; Alexa.getIntentName(handlerInput.requestEnvelope) ===
&#x27;GetAIResponseIntent&#x27;;\n    },\n    async handle(handlerInput) {\n
const { intent } = handlerInput.requestEnvelope.request;\n\n        // Log the
received intent and slots\n        console.log(&#x27;Received Intent:&#x27;,
intent);\n\n        // Check if the slot is filled\n        if
```

```
(!intent.slots.query.value) {\n            const speakOutput = &quot;I didn&#x27;t
catch that. Can you please ask me a question like, &#x27;What is quantum
mechanics?&#x27;&quot;;\n            console.log(&#x27;Slot value is empty.&#x27;);\n
return
handlerInput.responseBuilder\n                .speak(speakOutput)\n                .re
prompt(speakOutput)\n                .getResponse();\n        }\n\n        let
aiResponse = &#x27;&#x27;;\n        try {\n            // Log the query being sent to
OpenAI\n            console.log(&#x27;Sending query to OpenAI:&#x27;,
intent.slots.query.value);\n            aiResponse = await
getAIResponse(intent.slots.query.value);  // Call OpenAI API\n
console.log(&#x27;AI response received:&#x27;, aiResponse);\n        } catch (error)
{\n            console.log(&#x27;Error occurred while calling OpenAI API:&#x27;,
error);\n            aiResponse = &#x27;Sorry, I had trouble processing your
request.&#x27;;\n        }\n\n        // Return the AI response or error message\n
return
handlerInput.responseBuilder\n            .speak(aiResponse)\n            .getResponse
();\n    }\n};\n```\n\nThis will log critical parts of the interaction:\n- The
**intent** and **slot value** received by Alexa.\n- Whether the OpenAI API call was
made successfully.\n- The result of the OpenAI API call.\n- Any error that occurs
during the process.\n\n#### 2. **Test the OpenAI API Call**\nEnsure the **OpenAI API
call** is functioning correctly. Sometimes an unexpected response from OpenAI (e.g.,
malformed JSON) can crash your skill.\n\nHere&#x27;s a mock function you can use to
test your skill without making real API calls:\n\n```javascript\nasync function
getAIResponse(query) {\n    // Simulate a successful AI response for testing\n
return `This is a mock response for your query: ${query}`;\n}\n```\n\nYou can replace
your `getAIResponse` function with this mock version temporarily to see if the problem
is related to the OpenAI API call.\n\n#### 3. **Check for Timeouts or Lambda
Errors**\nIf your skill is running on AWS Lambda, there could be a timeout or other
Lambda-specific issues. Verify the following:\n- **Timeout Settings**: Ensure your
Lambda function has enough time to complete its execution. You can increase the
timeout in the AWS Lambda console.\n- **Lambda Logs**: Check the CloudWatch logs for
your Lambda function for any additional error information.\n\n#### 4. **Check for
Empty or Invalid Responses**\nMake sure that your skill is always returning a valid
response to Alexa. If there is a scenario where no response is sent (or the response
is empty), Alexa will return an `INVALID_RESPONSE`.\n\nEnsure you always return
something like this:\n```javascript\nreturn
handlerInput.responseBuilder\n    .speak(aiResponse)\n    .getResponse();\n```\n\nEven
if the OpenAI call fails, you should always return a response with a fallback message
to avoid triggering an error.\n\n### 5. **Inspect the Skill Metrics**\nFrom the logs,
you can see the `metrics` section which shows the **skillExecutionTimeInMilliseconds**.
If this value is very high, it could mean a timeout or slow response from the OpenAI
API.\n\n### Conclusion\n- **Add more logging** to see where the error occurs.\n-
**Mock the OpenAI API** to verify that your skill logic works without the API
dependency.\n- **Check for timeouts** and ensure a valid response is always returned
to Alexa.\n\nOnce you&#x27;ve added the additional logging and tested it, check the
logs in CloudWatch for more detailed insights. Let me know how it goes or if you need
more help!"]}, "status": "finished_successfully","end_turn": true
```