

-Team 6-

Jira - AI Integration



In partnership with:



Our Team

Let's meet our
team members

Mentors: **Mark Samuelson | Nir Peled**



GADI KOGEL



ARIEL HARRIS



NISIM COHEN



MOSHE EVYATAR GIDONI





What is JSM?

(Jira Service Management)

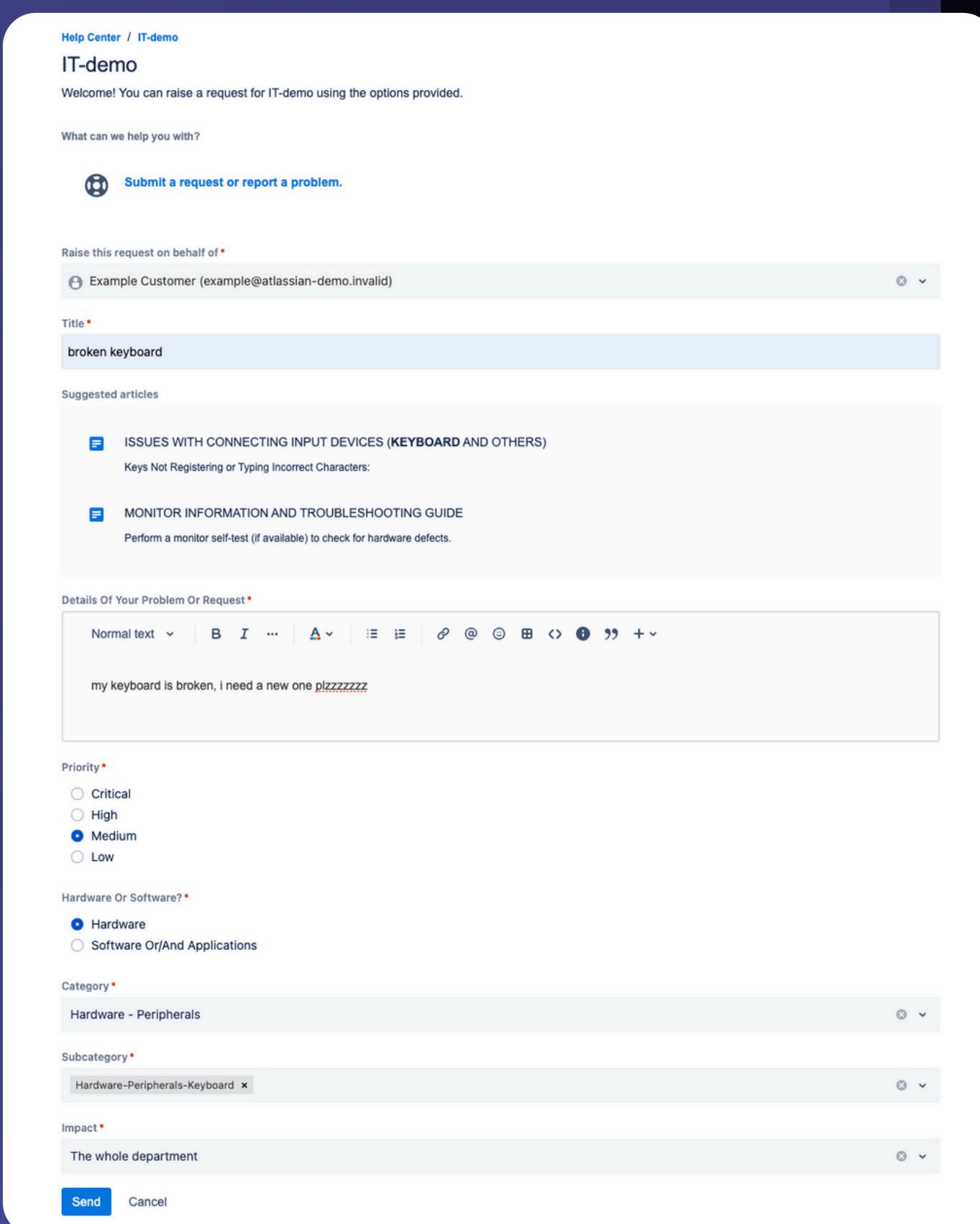
Jira Service Management (JSM) is a comprehensive platform designed to streamline IT service management processes and optimize communication between users and IT agents.

It offers a user-friendly interface for easy ticket submission, progress tracking, and collaboration. JSM features intuitive ticket submission, assignment to relevant IT agents, efficient communication tools, and robust reporting and analytics capabilities.

The screenshot shows the Jira Service Management (JSM) interface. At the top, there's a navigation bar with 'Jira', 'Your work', 'Projects', 'Filter', 'Dashboards', 'Teams', 'Apps', and 'Create'. Below the navigation is a search bar and a 'GROUP BY' dropdown set to 'None'. The main area is titled 'Projects / Beyond Gravity Board'. On the left, a sidebar menu includes 'Planning' (Timeline, Backlog, Board), 'Development' (Code, Releases), 'Project Pages', 'Add shortcut', and 'Project settings'. The 'Board' option is selected. The board itself has four columns: 'TO DO' (6 items), 'IN PROGRESS' (6 items), 'IN REVIEW' (6 items), and 'DONE' (6 items). Each column contains cards with ticket details like title, assignee, and status. For example, in the 'TO DO' column, there's a card for 'Optimize experience for mobile web' assigned to 'BILLING' with ticket ID 'NUC-344'. In the 'IN PROGRESS' column, there's a card for 'Fast trip search' assigned to 'ACCOUNTS' with ticket ID 'NUC-342'. The 'DONE' column has a card for 'Customers reporting shopping cart purchasing issues with the BG web store' assigned to 'ACCOUNTS' with ticket ID 'NUC-344'. A message at the bottom of the board says 'You're in a team-managed project' with a 'Learn more' link.



Ticketing system challenges



Often, the customer may not select the correct category that accurately reflects the issue or request for which they opened the ticket for.

This could waste some of the IT agent's time and impact how the data is analyzed, causing the BI to be less accurate.

Our Project Mission



• • •

1. to enhance ticket categorization.
2. to enhance prioritization.
3. to present quick and effective solutions.

Tackling the Task:

- Meeting with the IT team.
- Researching relevant software/platforms (Jira, OpenAI API).
- Brainstorming and developing a blueprint.

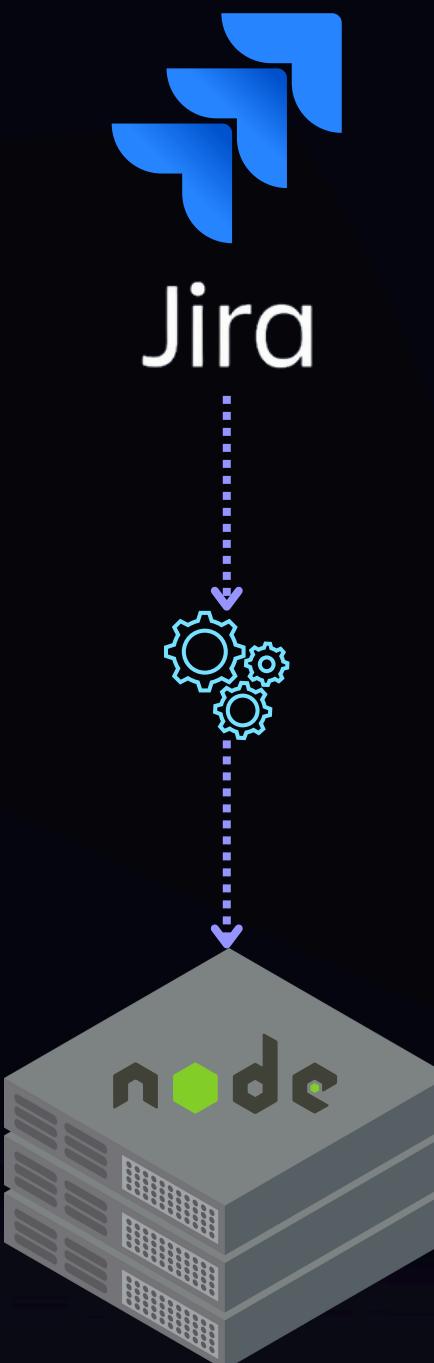




Demo Jira Project

We set up an Atlassian Cloud tenant with a demo Jira Service Management project designed to resemble AppsFlyer's current instance allowing us to safely explore Jira is a safe "sandbox" environment with "dummy data"

Automating the Ticket forwarding to the node.js



The next step was to automate sending tickets issued in Jira to a dedicated server. We created a webhook that triggers upon ticket creation, sending the ticket to the server for further handling.

Streamlining Ticket Processing with Node.js

A Node.js API was created to streamline ticket processing from Jira. This API serves as the project's core, handling HTTP requests and responses, ensuring stability.

```
...  
1 const express = require("express");  
2 const http = require("http");  
3 require("dotenv").config();  
4 require("./db/mongoConnect");  
5 const issuesRouter = require("./issuesRouter.routes");  
6  
7 // Initialize Express app  
8 const app = express();  
9  
10 app.use(express.json()); // Parse JSON bodies  
11 app.use("/", issuesRouter)  
12  
13 // Create HTTP server using Express app  
14 const server = http.createServer(app);  
15  
16 // Define port number  
17 const port = process.env.PORT || 3001  
18  
19 // Start server and listen for incoming requests  
20 server.listen(port, () => {  
|   console.log("Server is running on port", port);  
21 } )
```

Handling Tickets

Upon receiving a ticket, our API extracts pertinent fields and stores them in new objects. To prevent inadvertent transmission of sensitive data to our OpenAI assistant, we've devised a script using regex and custom functions to filter out such information (usernames, passwords, etc.), thereby ensuring data security.





OpenAI Assistant

- Our research focused on discovering the potential of the OpenAI API, primarily its AI assistant.
- A role has been created for the AI assistance, based on the ticket's Summary and Description the AI suggests the appropriate ticket category.
- Refined and optimized prompts to receive accurate categorizations, and deliver targeted next steps or solutions.

```
 1  const { OpenAI } = require('openai');
2  // Initialize conversation thread ID outside the function scope
3  let conversationThreadId;
4
5  exports.NewAiChat = async (issueData) => {
6      return new Promise(async (resolve, reject) => {
7          try {
8              // Check if issueData exists
9              if (issueData) {
10                  // Initialize OpenAI client
11                  const openai = new OpenAI({
12                      apiKey: process.env.OPENAI_API_KEY
13                  });
14
15                  // If there's no existing conversation thread, create a new one
16                  if (!conversationThreadId) {
17                      console.log("new conversation thread");
18                      const thread = await openai.beta.threads.create();
19                      conversationThreadId = thread.id;
20                  }
21
22                  // Function to add a message to the conversation thread
23                  async function addMessage(threadId, message) {
24                      const response = await openai.beta.threads.messages.create(
25                          threadId,
26                          {
27                              role: "user",
28                              content: message
29                          }
30                      );
31                      return response;
32                  }
33
34                  // Function to run the AI assistant on the conversation thread
35                  async function runAssistant(threadId) {
36                      const response = await openai.beta.threads.runs.create(
37                          threadId,
38                          {
39                              assistant_id: process.env.ASSISTANT_ID
40                          }
41                      );
42                      return response;
43                  }
44
45
46                  // Function to periodically check the status of the AI assistant
47                  async function checkingStatus(threadId, runId) {
48                      const runObject = await openai.beta.threads.runs.retrieve(
49                          threadId,
50                          runId
51                      );
52
53                      const status = runObject.status;
54
55                      if (status == 'completed') {
56                          // Once assistant run is completed, fetch messages from the thread
57                          const messagesList = await openai.beta.threads.messages.list(threadId);
58                          let messages = [];
59
60                          // Extract content from messages
61                          messagesList.body.data.forEach(message => {
62                              messages.push(message.content);
63                          });
64
65                          // Resolve the promise with AI response
66                          resolve(messages[0][0]?._text?.value);
67                      }
68
69                      // Send initial message to the AI assistant
70                      const message =
71                          `This is the object with the details:
72                          ${JSON.stringify(issueData)}
73                          `;
74
75                      await addMessage(conversationThreadId, message);
76
77                      // Start the AI assistant and periodically check its status
78                      const run = await runAssistant(conversationThreadId);
79                      const runId = run.id;
80
81                      // Periodically check the status of the AI assistant
82                      const pollingInterval = setInterval(() => {
83                          checkingStatus(conversationThreadId, runId);
84                      }, 5000);
85
86                  } catch (error) {
87                      // Reject the promise if any error occurs
88                      reject(error);
89                  }
90              }
91
92
93
94
95
96
97
98
99
99
```

Database

The ticket information and the AI feedback received are stored securely in MongoDB.

This setup enables the team to perform in-depth analysis, which empowers them to make well-informed decisions, improve existing workflows, and create effective and efficient new processes.



Example of Usage

The projector gives an error about my user information

Create subtask Link issue Add form Create ...

 Example Customer raised this request via Portal Hide details
[View request in portal](#)

Description
I tried to connect to the department's projector with username: EVI123 and with the password: 123456 and it gave me an error and didn't let me log in.
Would appreciate help...
Thanks!

Priority Medium

Hardware Or Software? Hardware

Category Hardware - Peripherals

Subcategory Other...

Impact The whole department

I want pizza

Create subtask Link issue Add form Create ...

 Moshe Evyatar Gidoni raised this request via Portal Hide details
[View request in portal](#)

Description
last night i dreeming about very big pizza with a lot of cheese and i want you to give me that

Priority Critical

Hardware Or Software? Hardware

Category Network

Subcategory Hardware-Peripherals-Keyb...

Impact The Whole Company

Returning to Jira

We've set up a system to add AI feedback as comments to Jira tickets, aiding administrators in quicker issue resolution. Additionally, it helps in reclassifying ticket categories and prioritizing requests, enhancing organizational efficiency.

 gadi 1 hour ago • Administrators

*Issue Key:  IB-236: The projector gives an error about my user information TO DO

*Issue Title: The projector gives an error about my user information

Suggested categories from the existing list

- Category: Hardware - Peripherals
- Subcategory: More.. - Audio / visual equipment

Priority and Impact Analysis

- Priority matches the content, as it affects departmental operations but is not immediately critical.
- Impact matches the content, assuming the projector's functionality is crucial for the whole department's activities.

Possible Solutions (Next Steps)

1. Verify the user credentials and ensure they have the correct permissions to access the projector.
2. Inspect the projector's connectivity and configuration settings for any misalignments or errors.
3. Arrange for a technical review or on-site support if the issue cannot be resolved remotely.

Edit · Delete · 

 gadi 44 minutes ago • Administrators

*Issue Key:  IB-232: I want pizza TO DO

*Issue Title: I want pizza

Suggested categories for creation

- Non-technical requests

Priority and Impact Analysis

- Priority does not match the content. Suggested priority: Low
- Impact does not match the content. Suggested impact: Only me

Possible Solutions (Next Steps)

1. Politely inform the customer this request is outside the scope of IT support.
2. Direct them to the correct department if applicable, or advise on proper channels for personal requests.
3. Update ticketing guidelines to clarify types of supported requests.

Edit · Delete · 

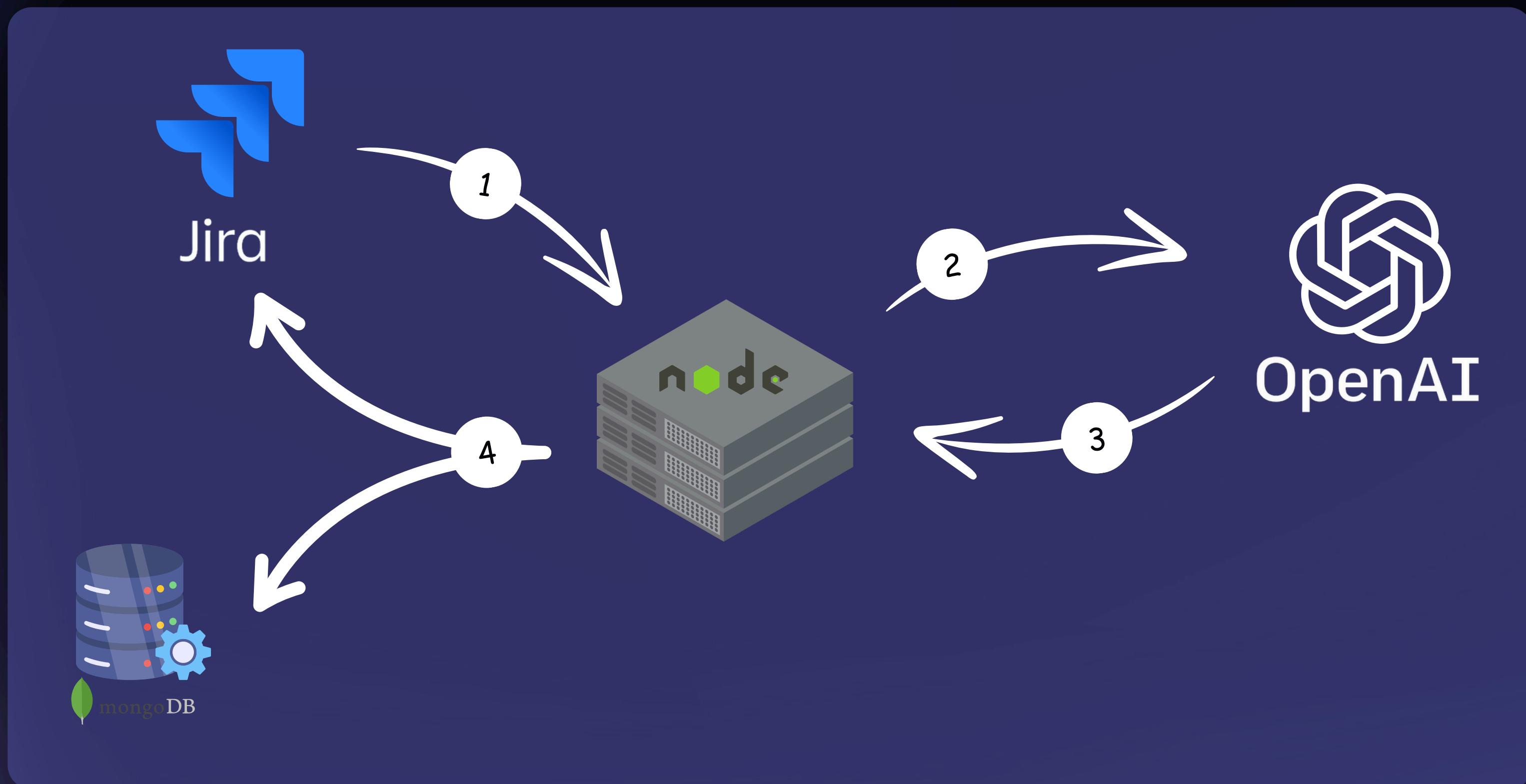
```
"_id": "663c8ee2af9781ae74737863",
"issueKey": "IB-232",
"issueTitle": "I want pizza",
"issueDescription": "last night i dreeming about very big pizza with a lot of cheese and i want you to give me that",
"priority": "Critical",
"category": "Network",
"subCategory": "Hardware-Peripherals-Keyboard",
"impact": "The Whole Company",
"aiFeedback": "***Issue Key:** IB-232\n**Issue Title:** I want pizza\n**Suggested categories for creation**\n- Non-  
- Non-technical requests\n- Priority and Impact Analysis\n- Priority does not match  
the content. Suggested priority: Low\n- Impact does not match the content. Suggested impact: Only me\n- Possible  
Solutions (Next Steps)\n1. Politely inform the customer this request is outside the  
scope of IT support.\n2. Direct them to the correct department if applicable, or advise on proper channels for  
personal requests.\n3. Update ticketing guidelines to clarify types of supported requests.",
"createdAt": "2024-05-09T08:52:50.567Z",
"updatedAt": "2024-05-09T08:52:50.567Z",
"_v": 0
```

```
"_id": "663c95c9af9781ae7473786b",
"issueKey": "IB-236",
"issueTitle": "The projector gives an error about my user information",
"issueDescription": "I tried to connect to the department's projector with username: *username* and with the password:  
*password* and it gave me an error and didn't let me log in. Would appreciate help... Thanks! ",
"priority": "Medium",
"category": "Hardware - Peripherals",
"subCategory": "Oter...",
"impact": "The whole department",
"aiFeedback": "***Issue Key:** IB-236\n**Issue Title:** The projector gives an error about my user  
information\n**Suggested categories from the existing list**\n- Category: Hardware - Peripherals\n- Subcategory: More.. - Audio / visual equipment\n- Priority and Impact Analysis\n- Priority matches  
the content, as it affects departmental operations but is not immediately critical.\n- Impact  
matches the content, assuming the projector's functionality is crucial for the whole department's activities.  
\n- Possible Solutions (Next Steps)\n1. Verify the user credentials and ensure they have  
the correct permissions to access the projector.\n2. Inspect the projector's connectivity and configuration  
settings for any misalignments or errors.\n3. Arrange for a technical review or on-site  
support if the issue cannot be resolved remotely.",
"createdAt": "2024-05-09T09:22:17.740Z",
"updatedAt": "2024-05-09T09:22:17.740Z",
"_v": 0
```



MongoDB®

Solution Flow



Product Value

1

AI-Driven Category Suggestions

Implementing AI to suggest new categories based on the frequency of similar issues submitted enhances the categorization process.

2

Reassessment of Ticket Priority and Impact

Regularly reassessing the priority and impact level of tickets to improve the efficiency and effectiveness of the IT department

3

Provision of Quick Resolutions

Offering quick and common solutions for recurring issues, thereby resolving tickets more efficiently.

4

Development of a Comprehensive Database:

Building a detailed database of ticket information and AI-generated comments to facilitate deeper analysis and continuous improvement.

Thank you!

Presented by:

Gadi kogel, Ariel Harris, Nisim Cohen and Moshe Evyatar Gidoni

• • •

Thanks

Ort Singalovski College

AppsFlyer

AppsFlyer's mentors:

Mark Samuelson & Nir Peled

IDF - Minhelet Charedim

DiversiTech - Avigail Michelson