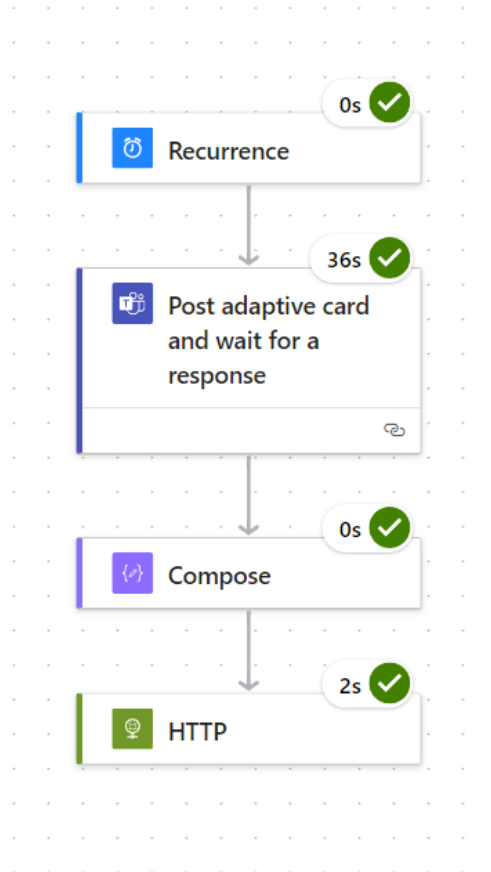


Teams Bot

Making the bot in Power Automate

I created a bot that is triggered once a day and sends a multiple-choice questionnaire to a Teams chat.



The post adaptive card action displays the questions to the user in Teams. The Compose action gets the answer data from the user. The HTTP action sends the answer data to the backend.

The data inside the adaptive card is saved like in this form:


```
"data": {
  "workPlace": "",
  "stress": "",
  "salary": "",
  "recommendation": ""
}
```

and is then collected by the Compose action.

Inputs *

 data ×

The questionnaire in Teams:

 Bäckman Marc via Workflows Yesterday 15.00

Kuinka tyytyväinen olet työympäristösi?

☒ Erittäin tyytymätön

☐ Tyytymätön

☐ Neutraali

☐ Tyytyväinen

☐ Erittäin tyytyväinen

Kuinka paljon stressiä koet töissä?

☒ Liian paljon

☐ Paljon

☐ Sopivasti

☐ Vähän

☐ Erittäin vähän

Kuinka tyytyväinen olet palkkaasi?

☒ Erittäin tyytymätön

☐ Tyytymätön

☐ Palkkani on sopiva

☐ Tyytyväinen

☐ Erittäin tyytyväinen

Kuinka todennäköisesti suosittelisit
yritystämme työpaikkana ystävällesi?

☒ En suosittelisi

☐ En osaa sanoa

☐ Suosittelisin

The backend for collecting the data

I made the backend with Python Flask. It collects the data that is sent from the HTTP action in Power Automate and will then send the data to a database. The backend is deployed at Microsoft Azure.

Filter for any field...

Subscription equals all

Resource group equals all ✕

+ Add filter

More (1)

Showing 1 to 1 of 1 records.

No grouping

List view

<input type="checkbox"/>	Name ↑↓	Status ↑↓	Location ↑↓	Pricing Tier ↑↓	App Service Plan ↑↓	Subscription ↑↓	App Type ↑↓
<input type="checkbox"/>	teams-bot-2024	Running	East US	Free	mabackma_asp_03...	Azure for Students	Web App ...

Backend repository: <https://github.com/mabackma/TeamsBot>

The pipeline

I made the pipeline directly from Azure Command-Line Interface (CLI), where I initially cloned the repository. I used these commands to make the pipeline with my repository:

Navigate to your App Service resource

```
az webapp show --name <app-name> --resource-group <resource-group-name>
```

Set up continuous deployment from GitHub

```
az webapp deployment source config --name <app-name>
--resource-group <resource-group-name> --repo-url <github-repo-url>
--branch <branch-name> --git-token <github-personal-access-token>
```

```
mabackma [ ~ ]$ az webapp deployment source config --name teams-bot-2024 --resource-group mabackma_rg_3183 --repo-url https://github.com/mabackma/TeamsBot.git --branch master --git-token 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
location is not a known attribute of class <class 'azure.mgmt.web.v2023_01_01.models._models_py3.SourceControl'> and will be ignored
source_control_name is not a known attribute of class <class 'azure.mgmt.web.v2023_01_01.models._models_py3.SourceControl'> and will be ignored
location is not a known attribute of class <class 'azure.mgmt.web.v2023_01_01.models._models_py3.SiteSourceControl'> and will be ignored
{
  "branch": "master",
  "deploymentRollbackEnabled": false,
  "githubActionConfiguration": null,
  "id": "/subscriptions/cf6cb00a-2818-4a71-b399-04f516d38869/resourceGroups/mabackma_rg_3183/providers/Microsoft.Web/sites/teams-bot-2024/sourcecontrols/web",
  "isGitHubAction": false,
  "isManualIntegration": false,
  "isMercurial": false,
  "kind": null,
  "location": "East US",
  "name": "teams-bot-2024",
  "repoUrl": "https://github.com/mabackma/TeamsBot",
  "resourceGroup": "mabackma_rg_3183",
  "type": "Microsoft.Web/sites/sourcecontrols"
}
```

The command explained:

“az webapp deployment source config”: We are configuring the deployment source for the web app

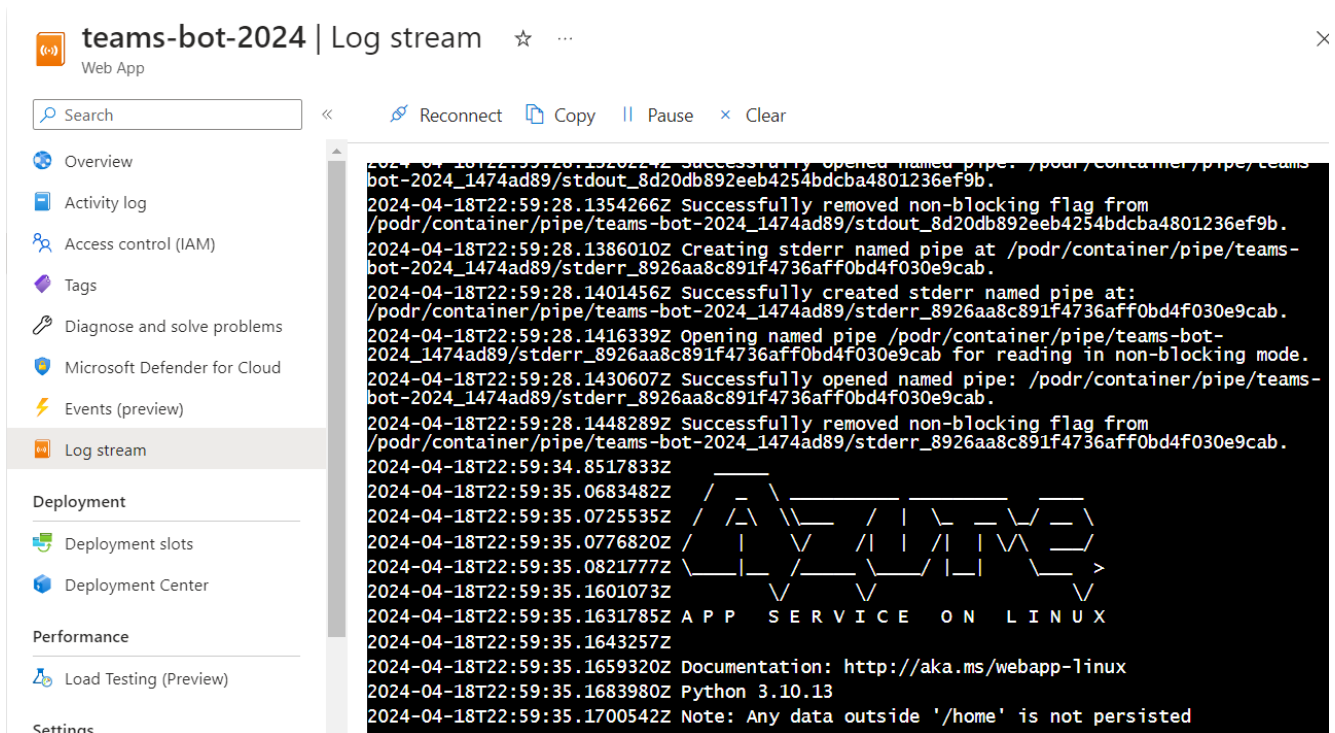
“—name teams-bot-2024”: the name of the Azure Web App

“—resource-group mabackma_rg_3183”: the name of the resource group (container that holds resources for the application).

“—repo-url <https://github.com/mabackma/TeamsBot.git>”: the GitHub address of the project.

“—branch master”: the branch used for deployment

I made a couple of changes to my repository and saw that the webapp in Azure rebooted after I pushed my changes to GitHub. The back end can be watched from looking at the log stream:

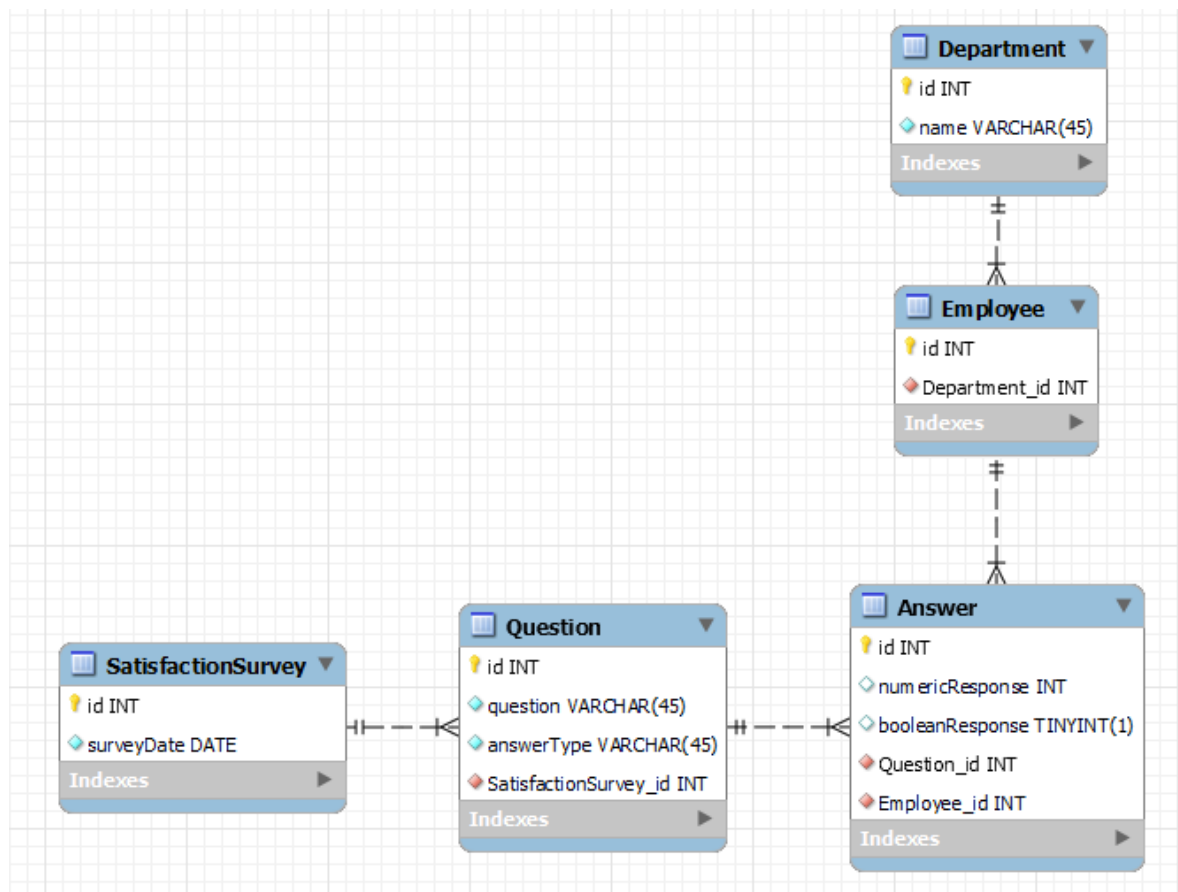


The screenshot shows the 'Log stream' interface for a web app named 'teams-bot-2024'. The left sidebar contains navigation links: Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Microsoft Defender for Cloud, Events (preview), Log stream (selected), Deployment, Deployment slots, Deployment Center, Performance, Load Testing (Preview), and Settings. The main area displays a log stream with timestamps and log messages. The messages include: 'Successfully opened named pipe', 'Successfully removed non-blocking flag from', 'Creating stderr named pipe at', 'Successfully created stderr named pipe at', 'Opening named pipe', 'Successfully opened named pipe', 'Successfully removed non-blocking flag from', and 'APP SERVICE ON LINUX'. The log stream also shows a ASCII art logo for 'APP SERVICE ON LINUX'.

```
2024-04-18T22:59:28.1354266Z Successfully opened named pipe: /podr/container/pipe/teams-bot-2024_1474ad89/stdout_8d20db892eeb4254bdcba4801236ef9b.
2024-04-18T22:59:28.1354266Z Successfully removed non-blocking flag from /podr/container/pipe/teams-bot-2024_1474ad89/stdout_8d20db892eeb4254bdcba4801236ef9b.
2024-04-18T22:59:28.1386010Z Creating stderr named pipe at /podr/container/pipe/teams-bot-2024_1474ad89/stderr_8926aa8c891f4736aff0bd4f030e9cab.
2024-04-18T22:59:28.1401456Z Successfully created stderr named pipe at: /podr/container/pipe/teams-bot-2024_1474ad89/stderr_8926aa8c891f4736aff0bd4f030e9cab.
2024-04-18T22:59:28.1416339Z Opening named pipe /podr/container/pipe/teams-bot-2024_1474ad89/stderr_8926aa8c891f4736aff0bd4f030e9cab for reading in non-blocking mode.
2024-04-18T22:59:28.1430607Z Successfully opened named pipe: /podr/container/pipe/teams-bot-2024_1474ad89/stderr_8926aa8c891f4736aff0bd4f030e9cab.
2024-04-18T22:59:28.1448289Z Successfully removed non-blocking flag from /podr/container/pipe/teams-bot-2024_1474ad89/stderr_8926aa8c891f4736aff0bd4f030e9cab.
2024-04-18T22:59:34.8517833Z
2024-04-18T22:59:35.0683482Z
2024-04-18T22:59:35.0725535Z
2024-04-18T22:59:35.0776820Z
2024-04-18T22:59:35.0821777Z
2024-04-18T22:59:35.1601073Z
2024-04-18T22:59:35.1631785Z APP SERVICE ON LINUX
2024-04-18T22:59:35.1643257Z
2024-04-18T22:59:35.1659320Z Documentation: http://aka.ms/webapp-linux
2024-04-18T22:59:35.1683980Z Python 3.10.13
2024-04-18T22:59:35.1700542Z Note: Any data outside '/home' is not persisted
```

The database

I created the database schema using mySQL Workbench and generated a mySQL query for creating the schema.



```

-- Schema teamsBotDatabase
-- Table `teamsBotDatabase`.`Department`
CREATE TABLE IF NOT EXISTS `teamsBotDatabase`.`Department` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE INDEX `id_UNIQUE` (`id` ASC),
  UNIQUE INDEX `name_UNIQUE` (`name` ASC))
ENGINE = InnoDB;

-- Table `teamsBotDatabase`.`Employee`
CREATE TABLE IF NOT EXISTS `teamsBotDatabase`.`Employee` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `Department_id` INT NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE INDEX `id_UNIQUE` (`id` ASC),
  INDEX `fk_Employee_Department1_idx` (`Department_id` ASC),
  CONSTRAINT `fk_Employee_Department1`
    FOREIGN KEY (`Department_id`)
      REFERENCES `teamsBotDatabase`.`Department` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- Table `teamsBotDatabase`.`SatisfactionSurvey`
CREATE TABLE IF NOT EXISTS `teamsBotDatabase`.`SatisfactionSurvey` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `surveyDate` DATE NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE INDEX `id_UNIQUE` (`id` ASC),
  UNIQUE INDEX `surveyDate_UNIQUE` (`surveyDate` ASC))
ENGINE = InnoDB;

-- Table `teamsBotDatabase`.`Question`
CREATE TABLE IF NOT EXISTS `teamsBotDatabase`.`Question` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `question` VARCHAR(45) NOT NULL,
  `answerType` VARCHAR(45) NOT NULL,
  `SatisfactionSurvey_id` INT NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE INDEX `id_UNIQUE` (`id` ASC),
  INDEX `fk_Question_SatisfactionSurvey_idx` (`SatisfactionSurvey_id` ASC),
  CONSTRAINT `fk_Question_SatisfactionSurvey`
    FOREIGN KEY (`SatisfactionSurvey_id`)
      REFERENCES `teamsBotDatabase`.`SatisfactionSurvey` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- Table `teamsBotDatabase`.`Answer`
CREATE TABLE IF NOT EXISTS `teamsBotDatabase`.`Answer` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `numericResponse` INT NULL,
  `booleanResponse` TINYINT(1) NULL,
  `Question_id` INT NOT NULL,
  `Employee_id` INT NOT NULL,
  UNIQUE INDEX `id_UNIQUE` (`id` ASC),
  PRIMARY KEY (`id`),
  INDEX `fk_Answer_Question1_idx` (`Question_id` ASC),
  INDEX `fk_Answer_Employee1_idx` (`Employee_id` ASC),
  CONSTRAINT `fk_Answer_Question1`
    FOREIGN KEY (`Question_id`)
      REFERENCES `teamsBotDatabase`.`Question` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Answer_Employee1`
    FOREIGN KEY (`Employee_id`)
      REFERENCES `teamsBotDatabase`.`Employee` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

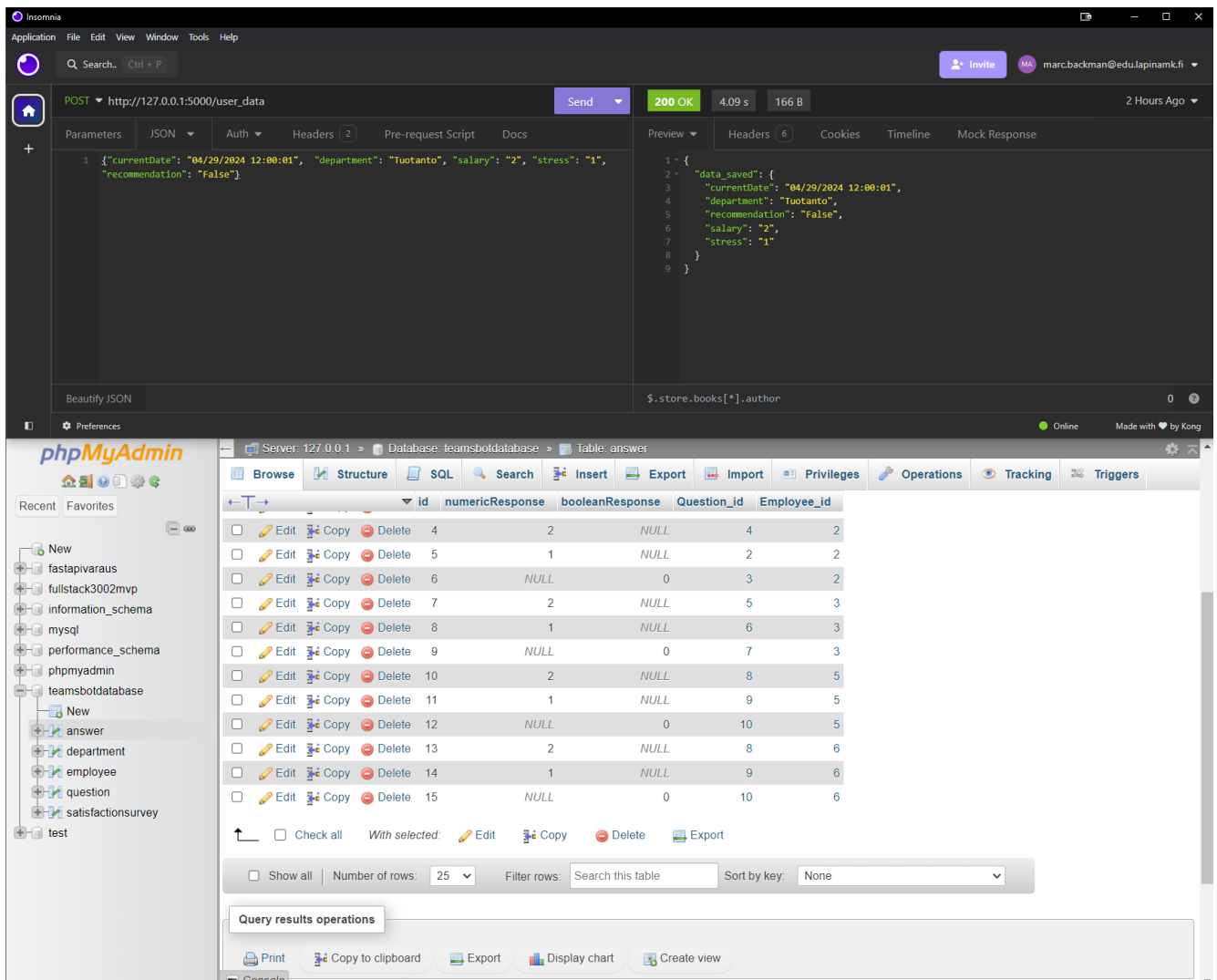
SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;

```

Testing the database

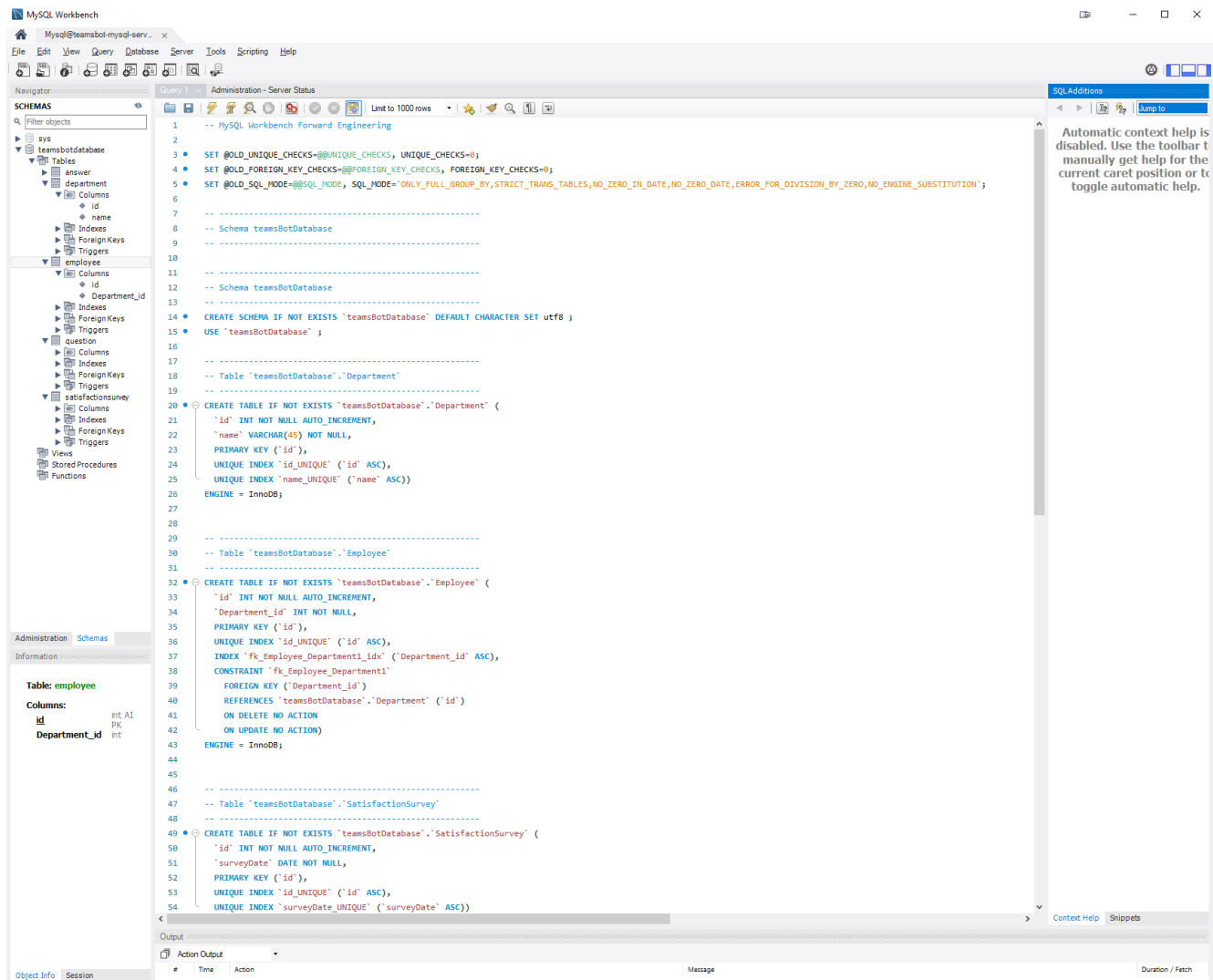
I tested the database with XAMPP and phpMyAdmin. This graphical user interface was useful to create the tables and see if my code would save data in the tables.

I made models for the tables in my mySQL schema to the Python Flask backend using SQLAlchemy. To test that my models were working correctly, I used Insomnia to send data to the backend and phpMyAdmin to see if it was saved in the database.



Creating the database in Azure

After verifying that everything between the backend and the mySQL database is working correctly, I created the database in a mySQL server in Azure through using mySQL Workbench.



I also added a connection string into my Python code for connecting to the MySQL database at Azure instead of the MySQL database at phpMyAdmin.

The connection string looks like this:

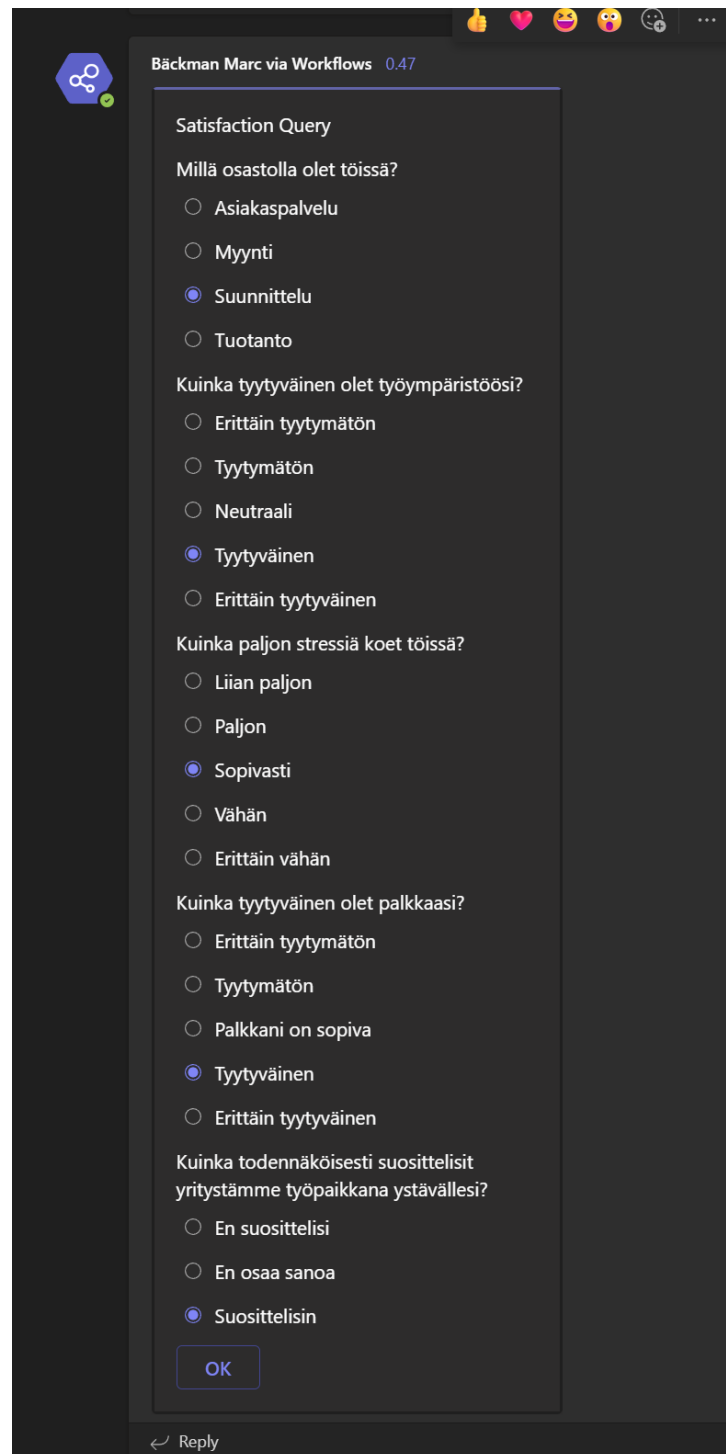
```
mysql+mysqlconnector://{azure_mysql_username}:{azure_mysql_password}@{azure_mysql_host}:{azure_mysql_port}/{azure_mysql_database}
```

I added environment variables for the connection string to the App service teams-bot-2024 in Azure.

Testing the workflow

I then tested the whole workflow by sending a survey from Power Automate to teams, answering the survey in teams, watching the log stream of the app service, and verifying the data saved in the MySQL database at Azure. Everything was working correctly.

In teams:



The screenshot shows a Microsoft Teams chat window. At the top, there's a header bar with a hexagonal icon on the left and a title bar on the right that says 'Bäckman Marc via Workflows 0.47'. The main content area contains a survey form titled 'Satisfaction Query'. The form has five sections, each with a question and four radio button options. The first section asks 'Millä osastolla olet töissä?' (In which department are you working?). The second asks 'Kuinka tyytyväinen olet työympäristösi?' (How satisfied are you with your work environment?). The third asks 'Kuinka paljon stressiä koet töissä?' (How much stress do you feel at work?). The fourth asks 'Kuinka tyytyväinen olet palkkaasi?' (How satisfied are you with your salary?). The fifth asks 'Kuinka todennäköisesti suosittelet yritystämme työpaikkana ystäväillesi?' (How likely are you to recommend our company as a workplace to your friends?). At the bottom of the form is an 'OK' button. Below the form, there's a 'Reply' button with a left-pointing arrow.

Satisfaction Query

Millä osastolla olet töissä?

- ☐ Asiakaspalvelu
- ☐ Myynti
- ☒ Suunnittelu
- ☐ Tuotanto

Kuinka tyytyväinen olet työympäristösi?

- ☐ Erittäin tyytymätön
- ☐ Tyytymätön
- ☐ Neutraali
- ☒ Tyytyväinen
- ☐ Erittäin tyytyväinen

Kuinka paljon stressiä koet töissä?

- ☐ Liian paljon
- ☐ Paljon
- ☒ Sopivasti
- ☐ Vähän
- ☐ Erittäin vähän

Kuinka tyytyväinen olet palkkaasi?

- ☐ Erittäin tyytymätön
- ☐ Tyytymätön
- ☐ Palkkani on sopiva
- ☒ Tyytyväinen
- ☐ Erittäin tyytyväinen

Kuinka todennäköisesti suosittelet yritystämme työpaikkana ystäväillesi?

- ☐ En suosittelisi
- ☐ En osaa sanoa
- ☒ Suosittelisin

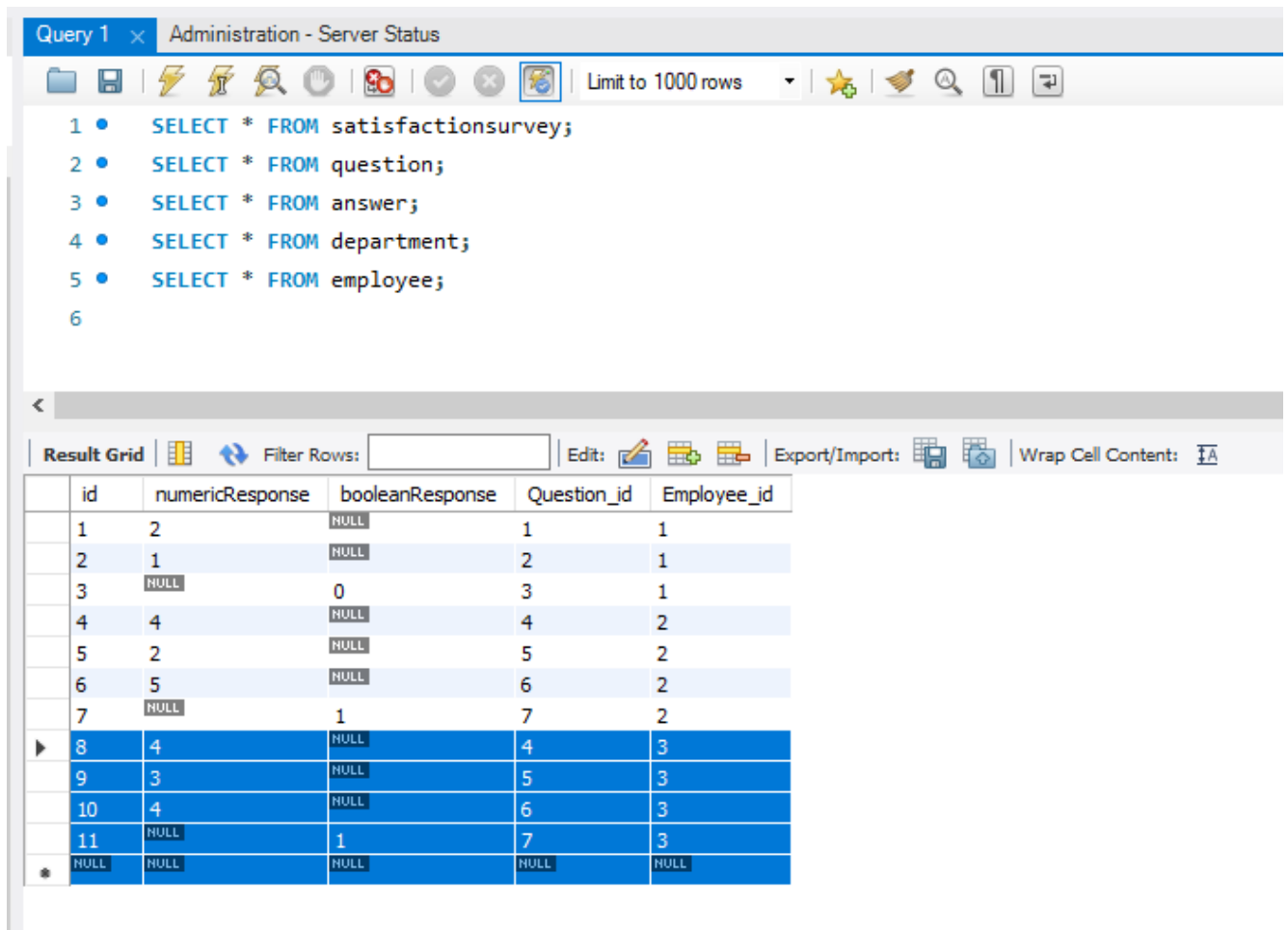
OK

Reply

After pressing OK button, in teams-bot-2024 log stream:

```
2024-04-28T21:46:41 No new trace in the past 16 min(s).
2024-04-28T21:47:41 No new trace in the past 17 min(s).
2024-04-28T21:48:41 No new trace in the past 18 min(s).
2024-04-28T21:49:29.3476579Z data received: {'currentDate': '04/28/2024 21:47:23',
'department': 'Suunnittelu', 'workPlace': '4', 'stress': '3', 'salary': '4',
'recommendation': 'True'}
2024-04-28T21:49:29.4027641Z 169.254.129.1 - - [28/Apr/2024:21:49:29 +0000] "POST
/user_data HTTP/1.1" 200 146 "-" "azure-logic-apps/1.0 (workflow
f191a87be1504345b7561f402a1b4015; version 08584877830800881858) microsoft-flow/1.0"
```


At mySQL database from making query at mySQL Workbench:



The screenshot shows the MySQL Workbench interface. The top toolbar includes icons for file operations, query execution, and a 'Limit to 1000 rows' dropdown. The query editor contains five SQL statements, each preceded by a numbered bullet point:

- 1 • `SELECT * FROM satisfactionsurvey;`
- 2 • `SELECT * FROM question;`
- 3 • `SELECT * FROM answer;`
- 4 • `SELECT * FROM department;`
- 5 • `SELECT * FROM employee;`
- 6

Below the query editor is the 'Result Grid' section. It features a 'Filter Rows' input field, an 'Edit' button, and 'Export/Import' and 'Wrap Cell Content' options. The result grid displays a table with the following data:

	id	numericResponse	booleanResponse	Question_id	Employee_id
	1	2	NULL	1	1
	2	1	NULL	2	1
	3	NULL	0	3	1
	4	4	NULL	4	2
	5	2	NULL	5	2
	6	5	NULL	6	2
	7	NULL	1	7	2
▶	8	4	NULL	4	3
	9	3	NULL	5	3
	10	4	NULL	6	3
	11	NULL	1	7	3
*	NULL	NULL	NULL	NULL	NULL

The workflow was tested a few times and then I concluded that it was working properly. I then started making a Streamlit application for displaying the data that was in the database. I should mention that I had to set the public ip address of my computer to the database in order to connect to it from mySQL Workbench.

Streamlit

I made a Streamlit application to show the results in different charts. Getting the results from the database had to be done through the backend service application that was deployed in Azure, because the mySQL database requires the public ip address for connecting to the database.

I wrote queries for getting data from the database. The queries were sent to the app service in Azure and then I made different Streamlit charts to display the data. I wanted to display data so that you could compare the answers from different departments. The nice thing about Streamlit charts is that they are interactive. For example, you can press on a department and remove it from the chart. Hovering your mouse over a department on a chart will also give you more information on the department.

The queries

Query all answers and department:

```
SELECT answer.id, department.name AS department_name
FROM answer
JOIN employee ON answer.employee_id = employee.id
JOIN department ON employee.department_id = department.id
```

Query answers over time with department and satisfactionsurvey:

```
SELECT satisfactionsurvey.surveyDate, department.name AS department_name
FROM answer
JOIN question ON answer.question_id = question.id
JOIN satisfactionsurvey ON question.satisfactionsurvey_id = satisfactionsurvey.id
JOIN employee ON answer.employee_id = employee.id
JOIN department ON employee.department_id = department.id
```

Query recommendation from everyone:

```
SELECT * FROM answer WHERE Question_id IN (SELECT id
FROM question WHERE question = 'recommendation' GROUP BY id)
```

Query average satisfaction with salary and department:

```
SELECT AVG(answer.numericResponse) AS avg_rating_salary, department.name AS department_name
FROM answer
JOIN employee ON answer.employee_id = employee.id
JOIN department ON employee.department_id = department.id
WHERE answer.Question_id IN (SELECT id FROM question
WHERE question = 'salary')
GROUP BY department_name
```

Query average satisfaction with work place and department:

```
SELECT AVG(answer.numericResponse) AS avg_rating_workplace, department.name AS department_name
FROM answer
JOIN employee ON answer.employee_id = employee.id
JOIN department ON employee.department_id = department.id
WHERE answer.Question_id IN (SELECT id FROM question
WHERE question = 'workPlace')
GROUP BY department_name
```

Query average satisfaction with stress and department:

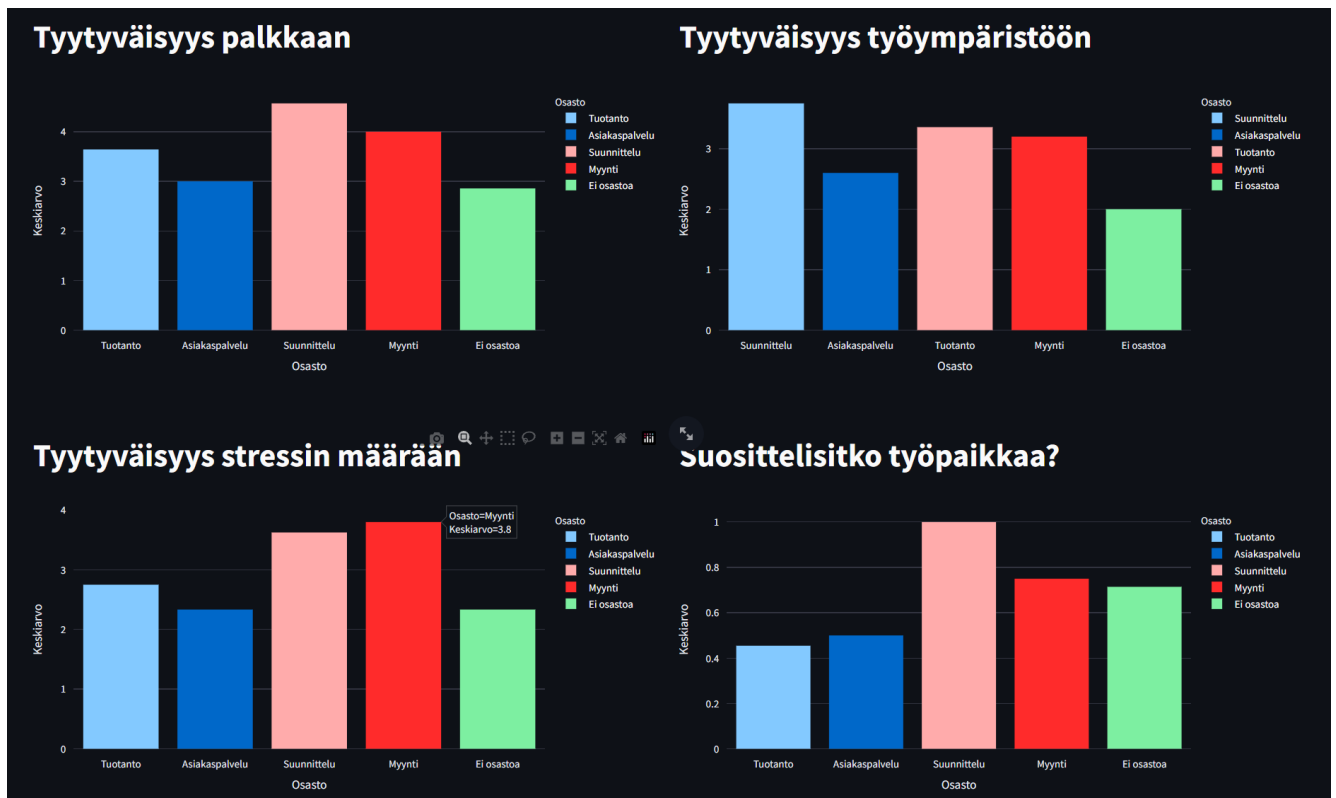
```
SELECT AVG(answer.numericResponse) AS avg_rating_stress, department.name AS department_name
FROM answer
JOIN employee ON answer.employee_id = employee.id
JOIN department ON employee.department_id = department.id
WHERE answer.Question_id IN (SELECT id FROM question WHERE question = 'stress')
GROUP BY department_name
```

Query average recommendation and department:

```
SELECT AVG(answer.booleanResponse)
AS avg_rating_recommendation, department.name
AS department_name
FROM answer
JOIN employee ON answer.employee_id = employee.id
JOIN department ON employee.department_id = department.id
WHERE answer.Question_id IN (SELECT id FROM question WHERE question = 'recommendation')
GROUP BY department_name
```

The charts





GitHub repositories:

Teams Bot app service: <https://github.com/mabackma/TeamsBot>

Streamlit application deployed at: <https://teams-bot.streamlit.app/>

Teams Bot Demo: <https://www.youtube.com/watch?v=TZwx1xhPjZI>

Conclusion

The workflow for the bot works as expected and the results are updated in a public Streamlit application. The Streamlit application is an effective way to monitor changes in the results in real time. However, the adaptive card for the survey should be sent to individual people in Teams instead of a channel. This is an easy fix that can be done in the Power Automate Flow.

Another improvement that could be done is using the NPS (Net Promoter Score) standard for a better calculation of the employees' satisfaction.

Marc Bäckman

REFERENCES:

Azure SQL Database Tutorial | Relational databases in Azure,
<https://www.youtube.com/watch?v=BgvEOkcR0Wk>

Develop applications with Azure Database for MySQL - Flexible Server,
<https://learn.microsoft.com/en-us/training/modules/develop-apps-with-azure-database-mysql/>