



Final Presentation

Brock University - Chatbot

Aamir, Alex, Matteo, and Mike

Table of contents

01

Scrum Process

Meetings, Sprints, Reviews

02

Infrastructure

Front and Backend architecture

03

Design

How are all the systems connected?

04

Implementation

What role does each system play in generating a response?

05

Technical Issues

Troubles and issues run into during the development process

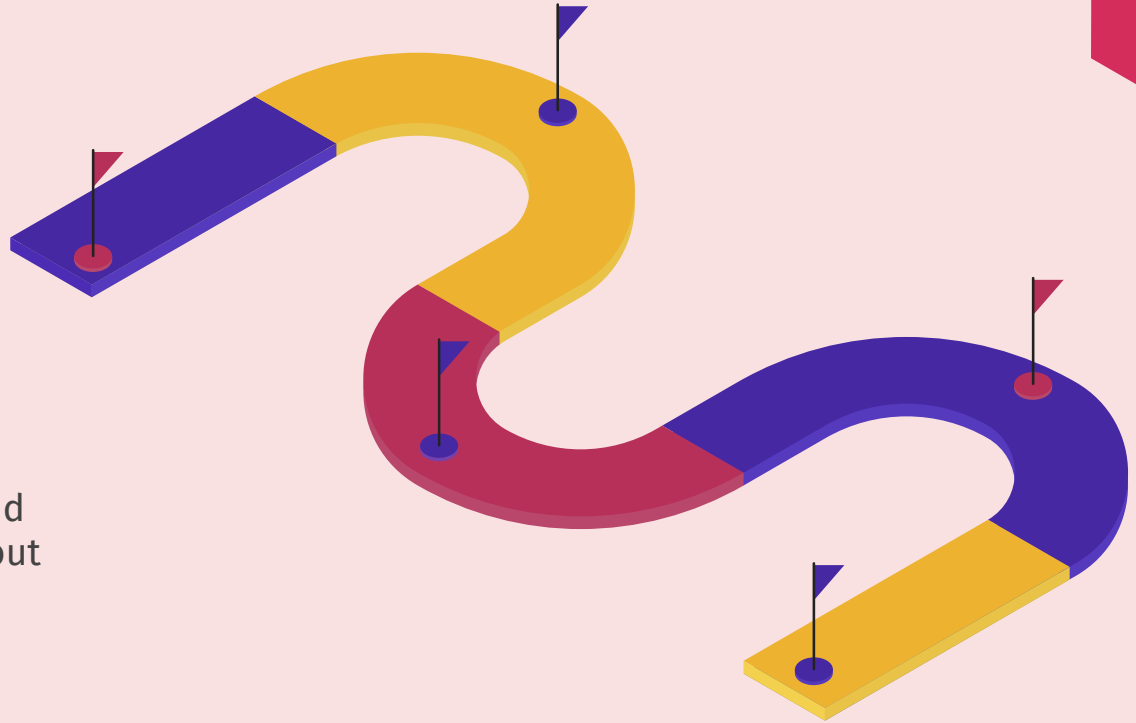
06

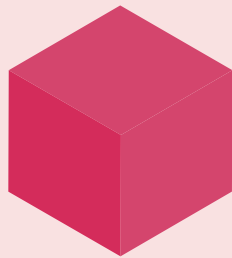
Demo

Chatbot demonstration

01 Scrum Process

How we reached our goal and made adjustments throughout each of our sprints.

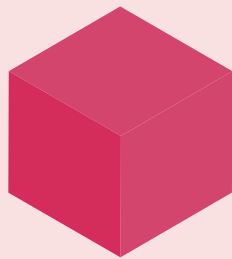




How we defined our sprints

- Our sprints were dependent on previous sprints
- The first sprint was meant to focus on foundational aspects of the project
- Subsequent sprints were modified to either have updated requirements or reduced/expanded tasks (depending on progress of previous sprints)
- Each sprint contained a mix of backend and frontend work

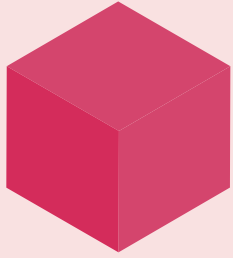




Product and Sprint Backlogs

Sprint	Goal
1	Concentrated on choosing foundational user stories for tasks (database creation and data gathering)
2	Concentrated on implementation of the backend structure for botpress and hosting of said backend
3	Focused on implementation of frontend
4	Concentrated on integration between the Botpress AI and the frontend GUI, as well as data gathering and training of the Chatbot
5	Worked on the testing and debugging the chatbot as well as merging the different versions for local Botpress instances.
6	Focused on adding in missing features, as well as more data gathering and training



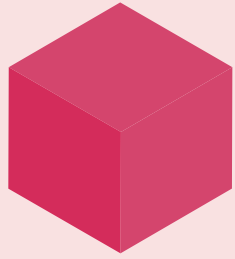


Meetings



- Bi-weekly Sprint Review meetings on Mondays at 4:50pm starting from February 7, 2022 onward.
- 30 minute Weekly Meetings on Thursdays at 6 p.m. starting from Jan 20, 2022 onward.

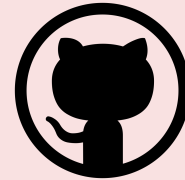
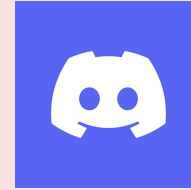


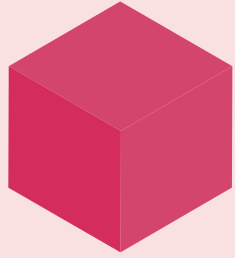


Tools we Used



Tool	Purpose
Discord	Used for our weekly meetings as well as a medium for quick discussion
MicroSoft Teams	Used for the bi-weekly Sprint Review meetings with the stakeholder
Github	Used as versioning control and ability to cohesively collaborate on the project
Jira	Used to create the Product/Sprint backlog for the project.



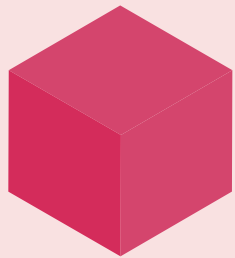


Progress Tracking



- Weekly meeting sync up
- Jira
- Github
- Feedback from stakeholders
- Mostly informal methods





Software Testing



System testing

- Botpress query/context testing
- Frontend system testing for Botpress API integration
- Frontend required feature testing and stress testing

Unit testing was not easily possible with Botpress API, however sample sentences were used to train the AI

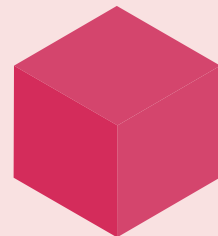


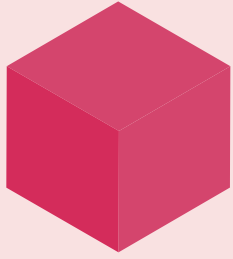


02

Infrastructure

AWS | Botpress Open Source | React





Data Collection

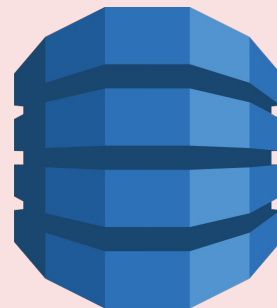




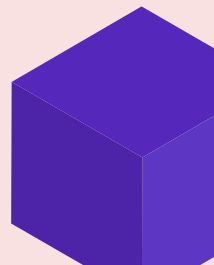
Data Scraping

Python

- API Calls
- BeautifulSoup
- Selenium Webdriver



Amazon RDS DB
Instance



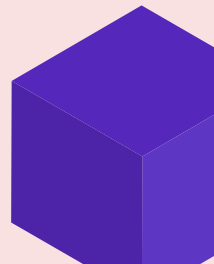


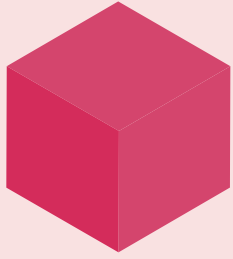
Database Tables

▼	courses
▼	Columns (16)
	code
	title
	description
	duration
	full_duration
	instructor
	location
	format
	restriction
	prereq
	exclusions
	notes
	days
	time
	type
	section

▼	exams
▼	Columns (7)
	code
	duration
	sectionid
	startday
	starttime
	endtime
	loc

▼	important_dates
▼	Columns (4)
	occasion
	session
	stakeholder
	date





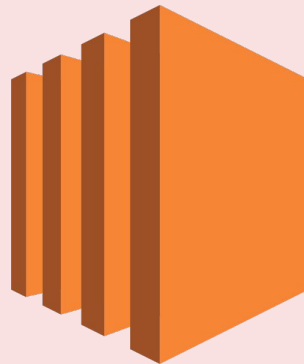
Chatbot





Chatbot - Botpress

- Is open source and available on github
 - Is allowed to be used commercially
 - Has a multitude of features which can be explored in the future.
- (Incorporation with other messaging tools)



botpress/botpress is licensed under the
GNU Affero General Public License v3.0

Permissions of this strongest copyleft license are conditioned on making available complete source code of licensed works and modifications, which include larger works using a licensed work, under the same license. Copyright and license notices must be preserved. Contributors provide an express grant of patent rights. When a modified version is used to provide a service over a network, the complete source code of the modified version must be made available.

This is not legal advice. [Learn more about repository licenses.](#)

Permissions

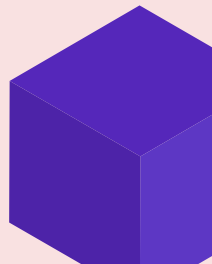
- ✓ Commercial use
- ✓ Modification
- ✓ Distribution
- ✓ Patent use
- ✓ Private use

Limitations

- ✗ Liability
- ✗ Warranty

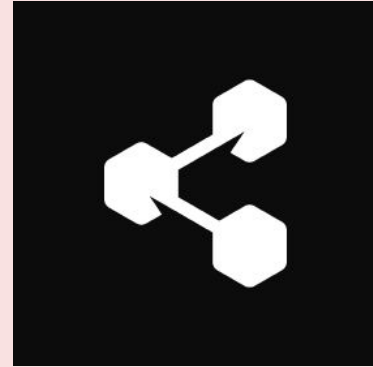
Conditions

- ① License and copyright notice
- ① State changes
- ① Disclose source
- ① Network use is distribution
- ① Same license



Why did we use Botpress?

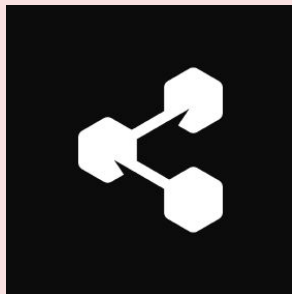
- Features a dashboard that can present many KPI's (Users, Most asked question, Etc)
- Chatbot can be modified/edited anywhere with appropriate access
- Comes with an efficient natural language processing unit
- Allows frontend connectivity through HTTP requests



How is Botpress setup?

2 Main things required when hosting

- ◆ Running the main program in a Docker container on an EC2.
- ◆ Storing Botpress data.
 - Hosted on a AWS RDS for scheduled backing up/redundancy.
 - The ability to connect with other instances of the program.

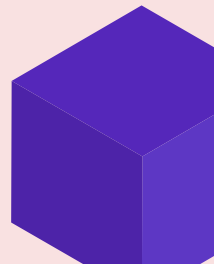
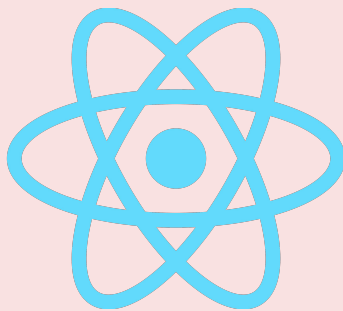




Frontend

Front End Development

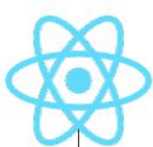
- Built using React.
- Hosted on Netlify
- Connects to Chatbot API



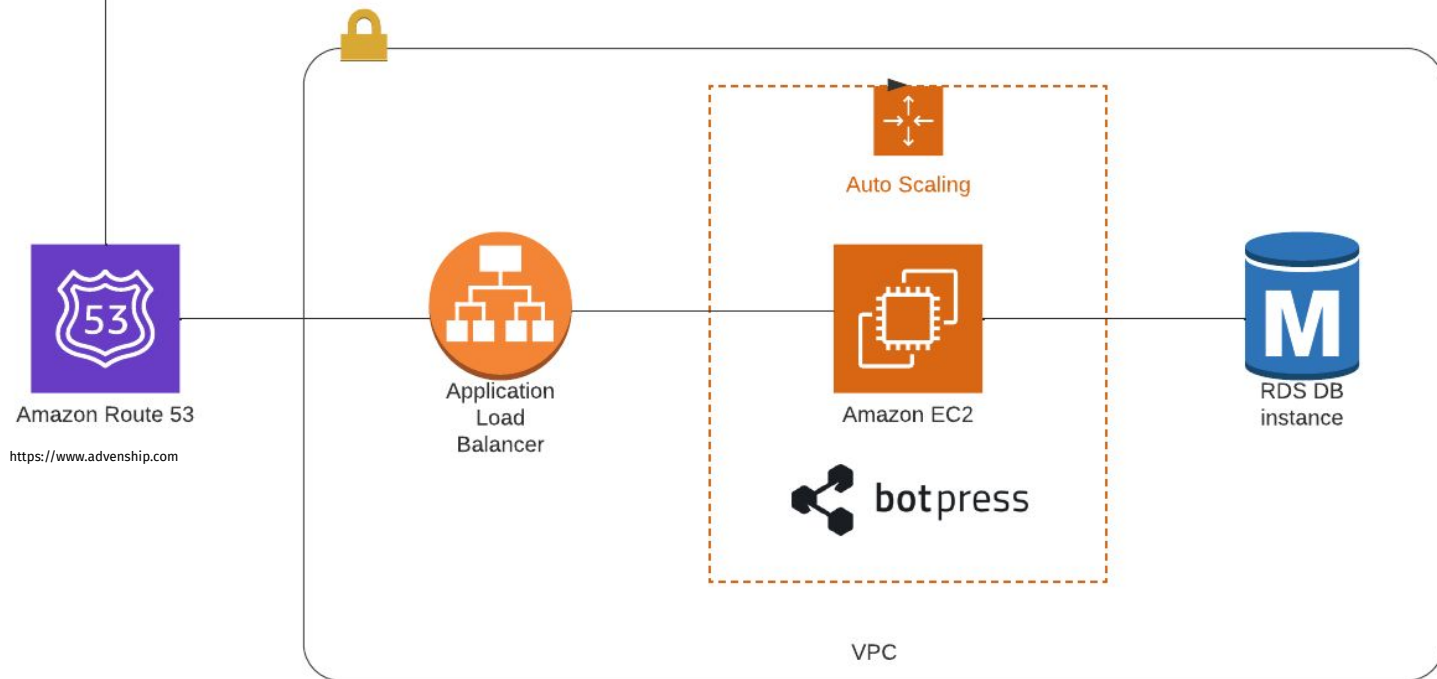


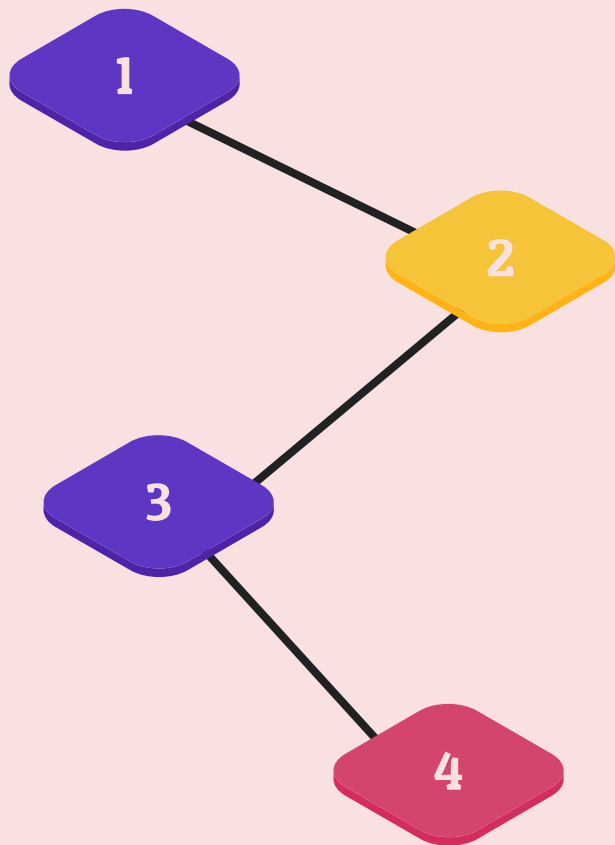
03 Design

Structure | Scalability | Security



netlify





04

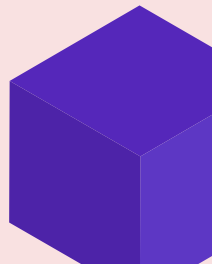
Implementation

How does all of this work in practice?



How does everything lead to answering a question?

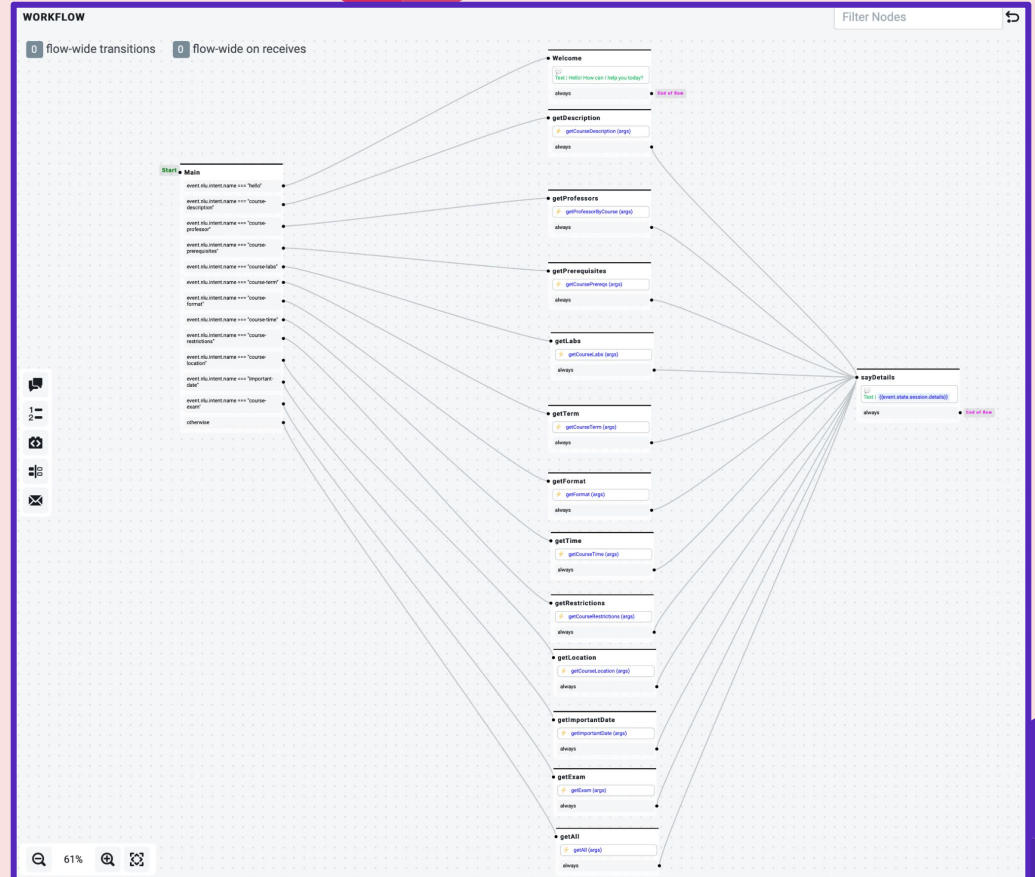
Let's follow an example





Chatbot Flow

‘When are labs for COSC 1P02?’

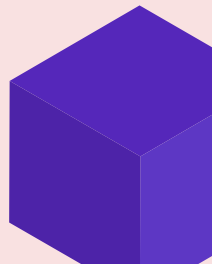




Intent Classification

**The first step is to figure out
what the users intent is**

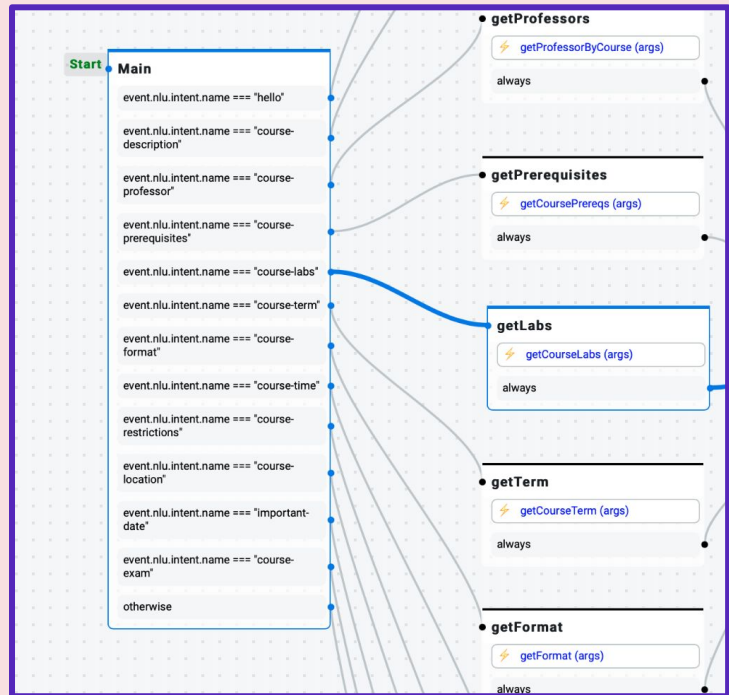
course-time
course-format
course-prerequisites
hello
course-professor
course-description
course-location
course-term
course-labs
course-exam
important-date
course-restrictions

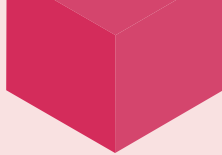




Intent Classification

As you can see the bot determined that the users intent was to ask about a courses labs





Intent Classification

The bot was 86.9% sure that the user was asking about labs

Top Intents

Entities

Slots

NLU		course-labs: 86.9 %	Type	Source	Normalized Value	Slot	Source	Extracted
NLU	course-exam: 13.1 %		keyPhrases	when	time	courseName	<u>COSC 1P02</u>	This turn
			keyPhrases	labs for	labs	occasion	<u>exams end</u>	2 turns ago
			dateOccasion	when are labs	Examinations begin			





Entity Extraction

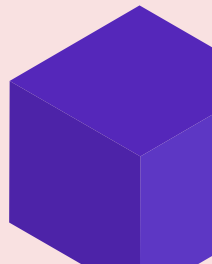
As we can see the bot was able to successfully select the course code based on the data it was trained with

Slots		
Slot	Source	Extracted
courseName	<u>COSC 1P02</u>	This turn



Entities possible
within a course-labs
intent

- 1 Are the lab options for COSC 4P02 .
- 2 labs for CHEM 2P90
- 3 does MATH 1p66 have labs
- 4 labs for biol 2f47
- 5 labs for entr 2p91
- 6 Are there labs for COSC 3p90
- 7 does chem 1F90 have labs
- 8 what days are labs for OBHR 2M87 on.
- 9 does actg 3N82 have labs.
- 10 Labs for MATH 1P67
- 11 Labs for MATH 1P66
- 12 labs LING 2p77
- 13 labs for ITIS 1p97
- 14 labs for ADED 1p31
- 15 are there labs for bchm on05
- 16 when are labs for apco 1p50
- 17 btgd 3p65 labs
- 18 chem 1p91 labs days
- 19 I need to know when the labs are for CHEM 2P20 ?
- 20 labs for cosc 1p03
- 21 labs for COSC 1p02
- 22 what are the labs for COSC 1P03
- 23 any labs for FREN 1p93
- 24 Does SPAN 2p95 have labs?



Receiving data from database

```
try {
  const client = new Client({
    user: 'masterbotaccess',
    host: 'botpressdb.ctor5sx6le6s.ca-central-1.rds.amazonaws.com',
    database: 'brockcollection',
    password: 'whyarefruitbowlssoexpensive',
    port: 5432
  })

  client.connect()

  const text =
    'SELECT * FROM COURSES top INNER JOIN ( SELECT duration, section, COUNT(section) AS numlabs FROM COURSES WHERE code LIKE $1 AND type LIKE $2 GROUP BY section, duration ) totalNum'
    ON top.section = totalNum.section AND top.duration = totalNum.duration WHERE code LIKE $1 AND type LIKE $2 ORDER BY top.duration, top.section, top.type'

  const values = [courseCode, 'LAB%']

  await client
    .query(text, values)
    .then(res => {
      if (Object.keys(res.rows).length === 0) {
        fullString = courseCode + ', ' + 'does not appear to have labs.'
      } else {
        var previousSection = ''
        var previousDuration = ''

```

Displaying Response

Since the format of the response is handled in the code while retrieving the data, all intents can use the same display mechanism



when are labs for COSC 1P02

COSC 1P02, Introduction to Computer Science, running in D2, has 19 labs in Section: 1.

LAB 1 runs Monday, from 800 -1000 @ SYNC

LAB 10 runs Friday, from 1500-1700 @ SYNC

LAB 11 runs Friday, from 1300-1500 @ SYNC

LAB 12 runs Wednesday, from 1300-1500 @ SYNC

LAB 13 runs Tuesday, from 1200-1400 @ SYNC

LAB 14 runs Tuesday, from 1400-1600 @ SYNC

LAB 15 runs Thursday, from 1200-1400 @ SYNC

LAB 16 runs Thursday, from 1400-1600 @ SYNC

LAB 17 runs Monday, from 1600-1800 @ SYNC

LAB 18 runs Thursday, from 1800-2000 @ SYNC



05

Technical Issues

Data Collection | Security | NLU Training



Data Collection

Goal

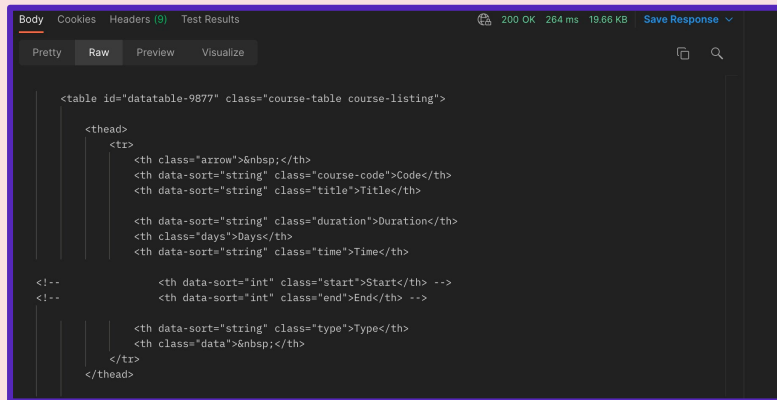
Make HTTP requests to the Brock Website directly

Issue

However that wasn't feasible with the rest of our solution because the response types received from the server

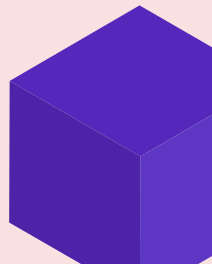
Solution

Settled on creating a data scraping script which placed all the data into a RDS



The screenshot shows the 'Raw' tab of a web browser's developer tools. The response is an HTML table with the following structure:

```
<table id="datatable-9877" class="course-table course-listing">
  <thead>
    <tr>
      <th class="arrow">&nbsp;</th>
      <th data-sort="string" class="course-code">Code</th>
      <th data-sort="string" class="title">Title</th>
      <th data-sort="string" class="duration">Duration</th>
      <th class="days">Days</th>
      <th data-sort="string" class="time">Time</th>
      <!-- <th data-sort="int" class="start">Start</th -->
      <!-- <th data-sort="int" class="end">End</th -->
      <th data-sort="string" class="type">Type</th>
      <th class="data">&nbsp;</th>
    </tr>
  </thead>
```





SSL Security

Goal

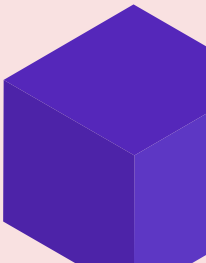

Make accessing the Botpress program secure (SSL)

Issue

There wasn't a sensible way of setting up the EC2 to only accept secure connections. NGINX was tried but wasn't cooperating with Docker Setup

Solution

Using Route 53 and Security groups to only allow connections from port 443





Training the NLU

Goal

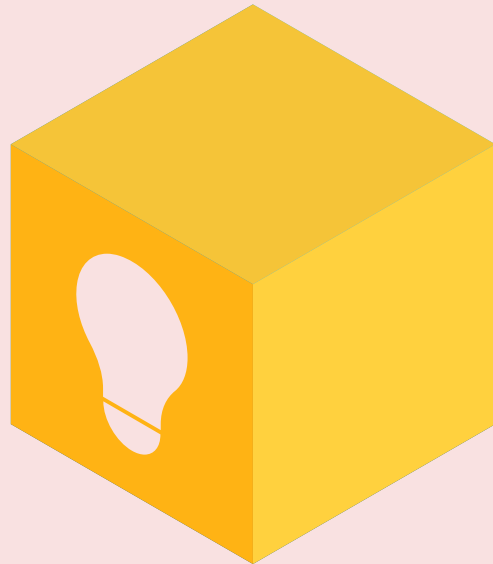
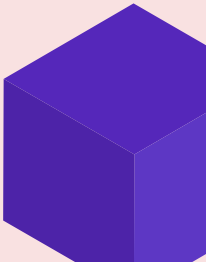

Apply changes that we made to the chatbot by training the NLU

Issue

Due to the specifications of the EC2 on AWS free tier we were unable to train the bot without crashing the system

Solution

Ran the program portion of Botpress on our local systems and connected to the RDS storing the Botpress data to train







06 Demo

<https://bu-bot.netlify.app>



Thanks!

Do you have any questions?



CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon** and infographics & images by **Freepik**