B.R.U.H Testing Document

Group 9, SENG 401 W25
Adam Yuen, Rodolfo Gil-Pereira, Mohammed Zaid Shaikh, Fahmi Sardar, Alexander Lai, Odin Fox

Testing and Validation Plan:

As we start our testing phase, based on our time constraints and choice of tech stack, these factors will heavily influence how we test our application. However, this should not undermine the importance of conducting tests properly. When taking into consideration time and our choice of programming language, developing test cases appears to be far less viable than an exploratory approach. Our choice of Python for our backend makes it more difficult to conduct unit tests due to its lack of emphasis on object-oriented programming, which makes it less viable for unit testing.

It is important to test in parallel to the development phase of our project when it comes to test-driven orientation. To properly conduct unit, integration, system, and acceptance testing, we will conduct exploratory testing as we build the application, testing individual functions, classes, or units before integrating them. This will look like comprehensively testing data flow through individual components; for instance, our proposed resume/cover letter classes will likely receive unit tests, and something, like our frontend react components, will be tested on various test inputs before they are slowly integrated with other react elements, and eventually the backend.

Validation will be conducted in a pair-programming method. Given that our exploratory testing will not be following as rigid of a testing regimen as other approaches, validating will be supplemented with the presence of another team member when empirical results are not as readily available. For instance, testing an API's JSON response will yield expected results, and validation can be conducted individually, but in the case of something like an LLM or text fields, where our data input and output often exhibit highly variable nature, another team member will be present during these tests to ensure that the results can be validated. In the case of more predictable elements of code, like our backend's Flask app, or the resume and cover letter classes, test cases and test suites are more applicable and thus will be utilized to make validation of these functionalities more concrete.

Exploratory Testing:

Approach details:

- Exploratory testing was conducted on both the front and backend portions of the application. This process was conducted alongside development to ensure that best practices outlined by the Test-Driven Orientation of testing were followed. Occasionally, our testing involved multiple developers in a pair-programming manner to improve the quality of our testing, oftentimes making our exploration of the code and UI more comprehensive and uncovering additional faults. Unit, regression, integration, and system testing were conducted to ensure all requirements were functional as the development continued. The records of these tests, as well as their charters, are included below.

Name(s)	**Adam Yuen, Zaid Shaikh**
Charter	Test the GPT API to ensure that prompts can be sent and received. Generate a basic prompt as a proof of concept so that future iterations will be able to use the function.
System Setup	- Windows 11 24H2 - Backend branch, isolated from frontend.
Test Start Time	8:58 PM 25/02/18
Areas Tested	- gptPromptingUtilities
How I Tested	Created a series of test prompts to determine how to best generate a cover letter using the GPT API to send and receive prompts. Cover letters were assessed by both testers to determine prompt effectiveness.
Defects	Nothing significant, however, switching the model from GPT3.5 to GPT4o appeared to yield better results
Actual vs Expected	The expected output was a prompt containing only the cover letter output. The function delivered what was expected.
Open Questions	N/A
Test Stop Time	9:16 PM 25/02/18
Other Non-Charter Testing	N/A

Name(s)	**Adam Yuen**
Charter	Test the application material class and determine if it

	performs its function correctly. This includes information extraction and prompt creation.
System Setup	- Windows 11 24H2 - Backend branch, isolated from frontend.
Test Start Time	12:23 PM 25/03/18
Areas Tested	- gptPromptingUtilities - applicationMaterials - Prompt templates in backend/llm folder
How I Tested	Made test JSON inputs in the ApplicationMaterial class to test the class's ability to create a prompt intended for the GPT prompter. Inputs and outputs for this can be found in the "coverLetterExample.py" and "resumeExample.py".
Defects	Inputting values into the constructor was implemented to support a string. Once implemented, this will not work because the front end will send it as a dictionary and not a string as originally intended.
Actual vs Expected	The expected outcome from the class tests is ultimately to make a prompt with the correct information that was extracted from a JSON. While this was ultimately accomplished, the defect raised in this test still raises a concern.
Open Questions	How does the frontend team want to submit data to the backend? Although JSON was agreed upon, what fields do they want to transmit to the backend?
Test Stop Time	12:56 PM 25/03/18
Other Non-Charter Testing	Tested sending the prompt to the GPT prompter to validate that prompts worked properly and the output is as desired.

Name(s)	**Adam Yuen, Zaid Shaikh**
Charter	Test the website API to ensure that data transfer between to the backend from the frontend will work before integrating the two things together.
System Setup	- Windows 11 24H2 - Backend branch, isolated from frontend.
Test Start Time	2:45 PM 25/03/19

Areas Tested	- gptPromptingUtilities - applicationMaterials - app.py
How I Tested	In a pair programming manner, we conducted tests on the backend using postman to simulate fetch requests to the backend before connecting the two layers of the application. Sample JSON provided by the frontend team was used in the Postman interface to validate the functionality of things. This can be found in the API tests.
Defects	In our prompts, handling the JSON received from the front end was not handled properly in some cases. Some of the fields that were supposed to be included in the prompts were not being delivered in the JSON package from the front end.
Actual vs Expected	The expected output can be found in the API tests. The test yielded the expected outcomes of our desired JSON responses.
Open Questions	What other data needs to be sent from the frontend, and how does the backend need to handle it?
Test Stop Time	3:42 PM 25/03/19
Other Non-Charter Testing	Revalidated the functionality of the ApplicationMaterial class now that it is integrated with the backend using regression testing.

Name(s)	**Adam Yuen**
Charter	Test the website's ability to collect material from the fields, package them, and send them to the backend to be processed.
System Setup	- Windows 11 24H2 - Backendbranch with frontend branch merged into it.
Test Start Time	8:42 PM 25/03/19
Areas Tested	coverletter.js and resume.js were tested.
How I Tested	By replacing the API endpoints for the react components to use my local instance of the backend to test it before deployment.

	Fields on the front end were filled out with typical data entered in, like typical names, different ways of entering job descriptions, and different job descriptions obtained from LinkedIn. The docker terminal where the backend was containerized on my machine was used to determine where errors went wrong.
Defects	Many fields were missing from the front end due to miscommunication between our teams. However, these were little details; the fundamental features required on the front and backends were implemented, meaning the faults that we found were not overly significant.
Actual vs Expected	Similar to the previous exploratory test, the expected outputs can be found in the API tests section
Open Questions	How can we improve communication efforts between the two teams to avoid this from happening again?
Test Stop Time	9:32 PM 25/03/19
Other Non-Charter Testing	N/A

Name(s)	**Alexander Lai**
Charter	Test the ability to view saved documents from a premade account and that the website does not crash when viewing from a new account.
System Setup	- Windows 11 24H2 - Locally run frontend - AWS Deployed backend - Deployed Mongo DB
Test Start Time	11:04 AM 25/03/20
Areas Tested	- App.js - Loginpage.js - Signuppage.js
How I Tested	I logged in with a pre-made test account and checked for saved documents. I then signed up with a new account. I then logged in with the new account and checked that there were no saved documents.
Defects	Page breaks if the name of a document is a JSON but by changing the default setting of the title to a string, the

	defect can be fixed.
Open Questions	What happens if the name input is formatted like a JSON?
Test Stop Time	12:30 PM 25/03/20
Other Non-Charter Testing	Tested the ability to save a document on a new account and view it.

Name(s)	**Fahmi Sardar**
Charter	Verify seamless navigation and functionality of the Login and Signup pages, ensuring all interactive elements operate as intended and provide a smooth user authentication experience.
System Setup	- Windows 11 24H2 - Locally run frontend
Test Start Time	7:34 PM 25/03/17
Areas Tested	- Loginpage.js - Signuppage.js
How I Tested	I logged into pre-existing accounts, created new accounts, and proceeded as a guest to verify proper navigation between the login and signup pages. Upon successful account creation or user authentication, I ensured that the user was correctly redirected to the homepage to proceed with their next action.
Defects	If the user enters incorrect credentials on the login page, there is no indication of which specific credential is incorrect
Open Questions	N/A
Test Stop Time	8:00 PM 25/03/17
Other Non-Charter Testing	N/A

Name(s)	**Fahmi Sardar**

Charter	Ensure the correct visual highlighting, state activation of buttons and proper error messages in **ViewSaved.js**, validating their responsiveness and intended functionality.
System Setup	- Windows 11 24H2 - Locally run frontend
Test Start Time	6:30 PM 25/03/20
Areas Tested	- ViewSaved.js - Resume.js - CoverLetter.js - Output.js
How I Tested	To initiate this test, I logged into a pre-existing account containing saved resumes and cover letters. I systematically navigated through the saved items, clicking buttons sequentially from left to right, verifying that the selected item's button remained highlighted to indicate the active selection. I then proceeded to select resumes and cover letters in varying sequences to ensure that the order of selection did not affect the button highlighting, maintaining a clear and consistent visual indicator for the user. Next, I created a new account to evaluate how **ViewSaved.js** would display an empty state with no resumes or cover letters. I confirmed that an appropriate message was displayed, informing the user that no items had been created. To further validate the system's behavior, I generated three resumes and three cover letters, then navigated back to the **View Saved** page to ensure all six entries appeared correctly. I proceeded to delete each entry individually, verifying that the same notification message reappeared once no saved items remained. Finally, I logged out and attempted to access the **View Saved** page as a guest. This step ensured that users who are not signed in receive a clear notification informing them that authentication is required to view saved resumes and cover letters.
Defects	The error message box and the preview box are built upon a shared code structure, differentiated only by conditional logic. However, an intermittent issue arises where, following the deletion of an entry, the preview box momentarily persists before seamlessly transitioning to the error message box. This brief overlap may cause a

	visual delay, impacting the clarity of the user experience.
Open Questions	N/A
Test Stop Time	9:00 PM 25/03/20
Other Non-Charter Testing	This exploratory test further validated that both the **Resume** and **Cover Letter** pages correctly redirected the user to **Output.js**, ensuring seamless navigation and proper functionality of the save feature. Additionally, the test confirmed that the logout mechanism operated as intended, effectively terminating the user session and preventing unauthorized access to ViewSaved.js

Name(s)	**Zaid Shaikh**
Charter	Ensure that the functionality of downloading a docx file in the “Generation Output” page is properly functional.
System Setup	- Windows 11 24H2 - Locally run frontend
Test Start Time	5:30 PM 25/03/21
Areas Tested	- Output.js
How I Tested	This test was performed by running the website on local host port 3000 using “npm start”. I then navigated to the “Generation Output” page using the navigation bar located at the top of the website. I began to hit the “Download DOCX” button with no generated output and observed that a runtime error occurred due to no data present in the output field.
Defects	When attempting to click the “Download DOCX” button when no resume or cover letter has been generated, an error occurs in the website as no proper error check has been included to handle that scenario.
Actual vs Expected	The expected outcome was that an error message would appear in the form of an alert or a direct error message displayed on the website. The actual result was that the JavaScript file displayed a runtime error.
Open Questions	N/A
Test Stop Time	5:45 PM 25/03/21

Other Non-Charter Testing	N/A

Acceptance Testing:

Name(s)	**Odin Fox**
Charter	Test the user experience and user interface to verify the software meets all requirements
System Setup	- Windows 11 24H2 - AWS deployed frontend and backend
Test Start Time	11:30 AM 25/03/21
Areas Tested	Entire system
How I Tested	Testing was performed via inputting several different sets of credentials in order to check for any variance or bias in LLM models or credential validation. Login and Signup were tested for correct validation of credentials to make sure passwords and emails were correctly being taken. Testing of the UI was being done to ensure consistent functionality. Resume and Cover Letter creation were tested via the same methodology of Login and Signup, inputting several sets of credentials, with and without certain fields filled.
Findings	-"Invalid Username or Password" text was not red and was hard to see in the first place -Error message doesn't show when "Password" and "Confirm Password" fields don't match
Actual vs Expected	These tests yielded the expected results without any major deviations.
Open Questions	N/A
Test Stop Time	12:32 PM 25/03/21
Other Non-Charter Testing	N/A

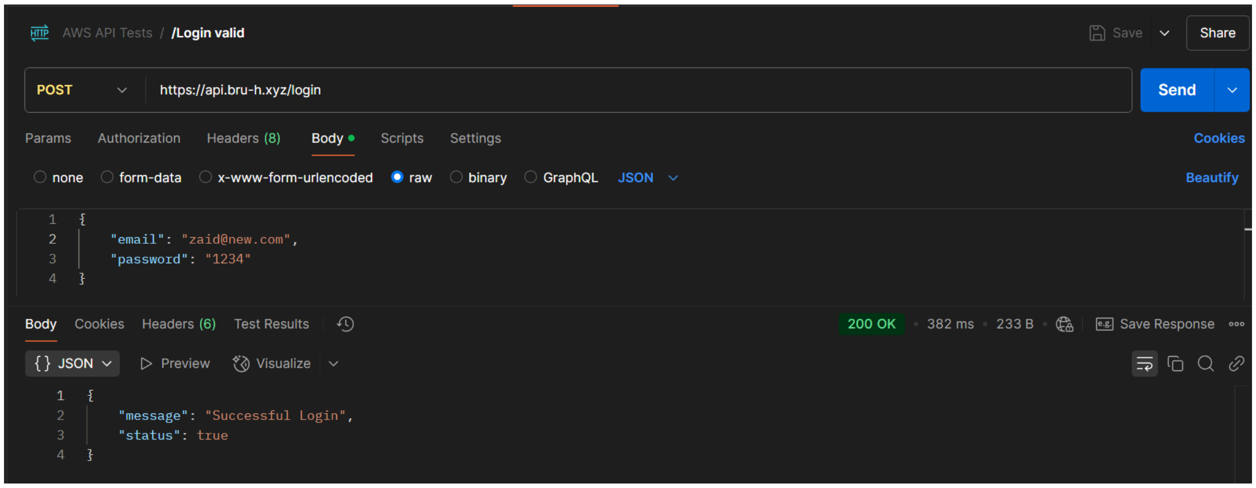
API Testing:

Approach:

- Postman was used to generate tests for many of the backend and frontend components before connecting them together, an important step in integration testing. Enclosed are the tests we utilized to ensure different API functionalities functioned as intended before integrating them. These tests also serve as a layer of regression testing; in particular, the Postman tests centered around testing prompts, and their output provided another layer of testing for these features. Features regarding login, signup, saving and retrieving user documents also account for database testing. Ensuring the backend is connected to the database and all is working correctly with it.

- The following postmans show the input body field and the output. All showing the expected output and behavior.
- Proper error handling occurs when inputs are wrong or missing

Login:



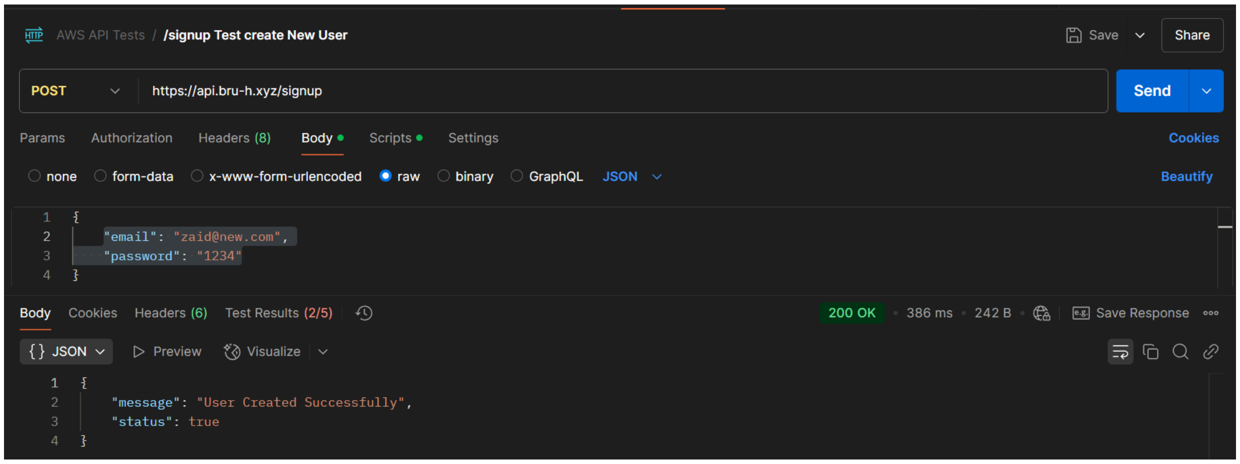
POST https://api.bru-h.xyz/login

```
{
    "email": "zaid@new.com",
    "password": "1234"
}
```

200 OK · 382 ms · 233 B

```
{
    "message": "Successful Login",
    "status": true
}
```

Sign Up (new user accepted)



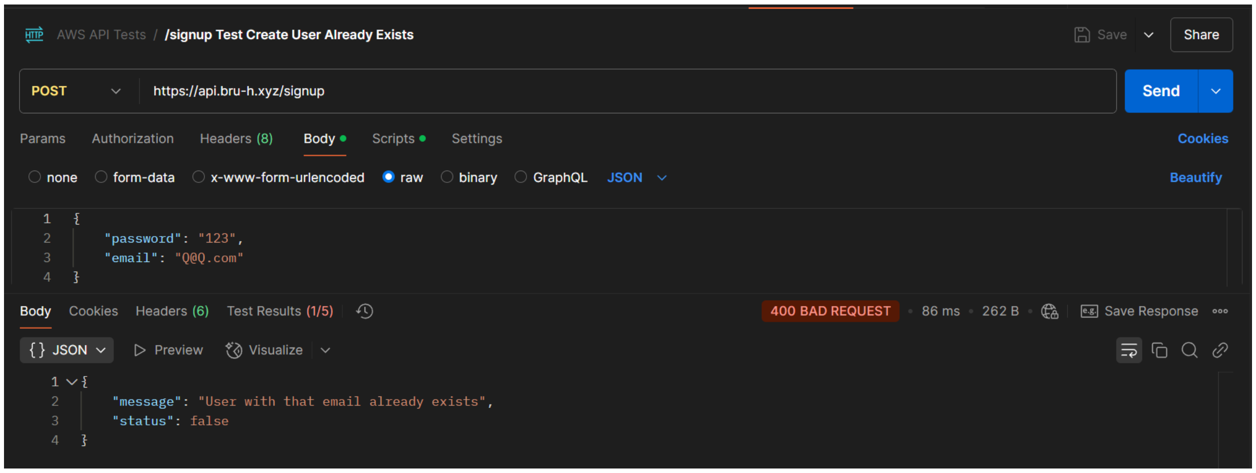
POST https://api.bru-h.xyz/signup

```
{
    "email": "zaid@new.com",
    "password": "1234"
}
```

200 OK · 386 ms · 242 B

```
{
    "message": "User Created Successfully",
    "status": true
}
```

Sign up (user already exists error)


AWS API Tests / /signup Test Create User Already Exists

POST https://api.bru-h.xyz/signup

Body: raw JSON

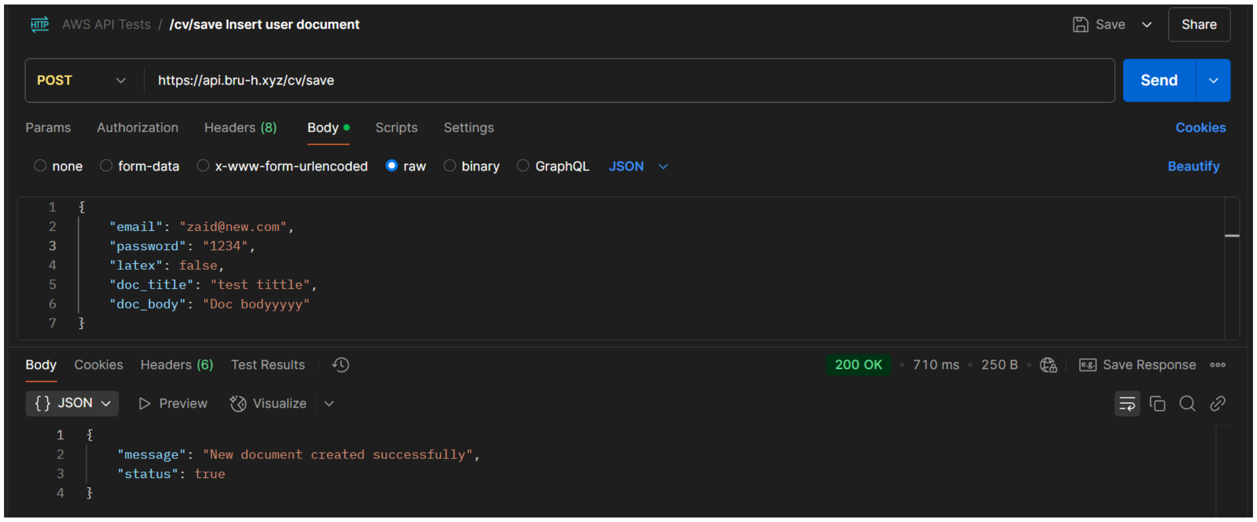
```
{
    "password": "123",
    "email": "Q@Q.com"
}
```

400 BAD REQUEST · 86 ms · 262 B

Test Results (1/5)

```
{
    "message": "User with that email already exists",
    "status": false
}
```

Create new document under a given user:


AWS API Tests / /cv/save Insert user document

POST https://api.bru-h.xyz/cv/save

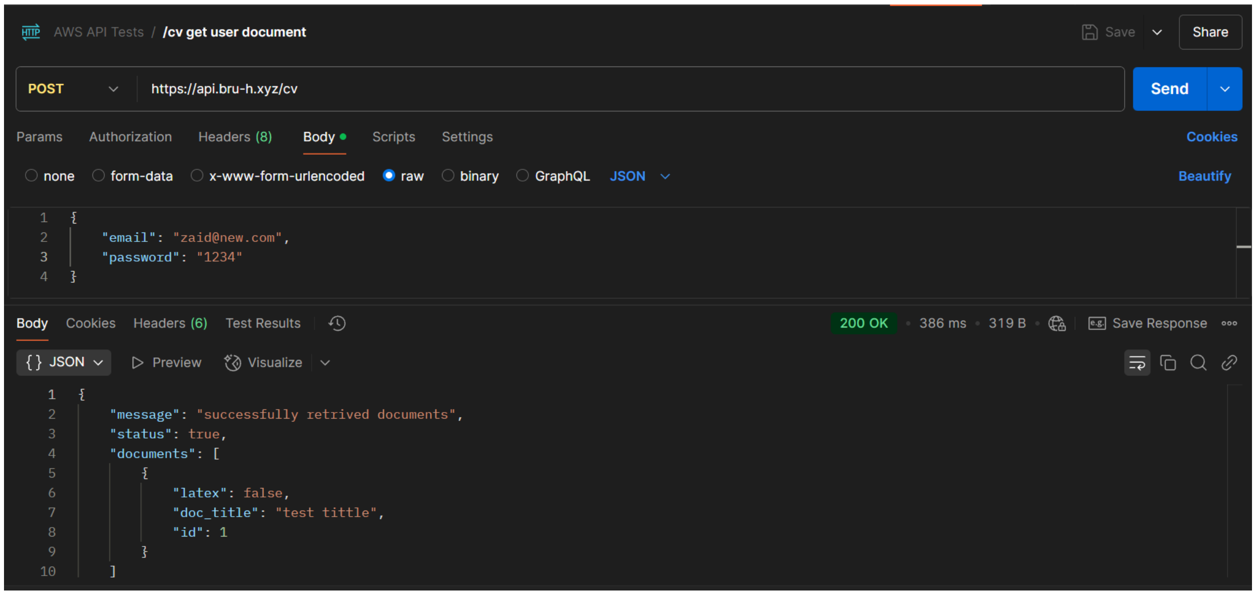
Body: raw JSON

```
{
    "email": "zaid@new.com",
    "password": "1234",
    "latex": false,
    "doc_title": "test tittle",
    "doc_body": "Doc bodyyyyy"
}
```

200 OK · 710 ms · 250 B

```
{
    "message": "New document created successfully",
    "status": true
}
```

Retrieve user documents:



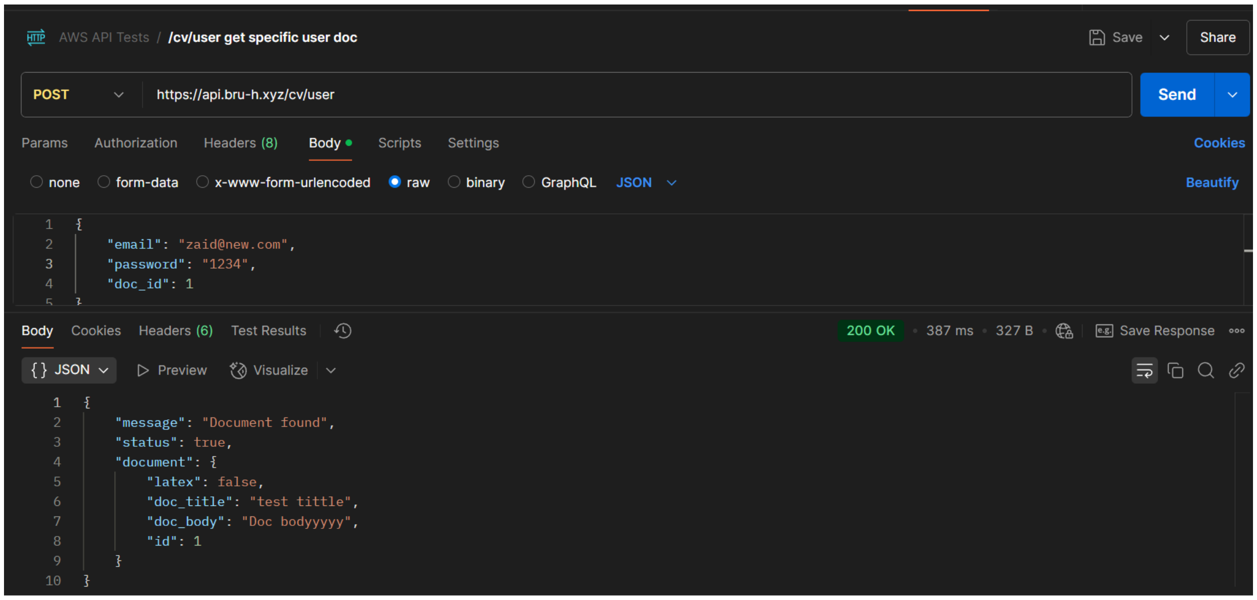
POST https://api.bru-h.xyz/cv

```
{
    "email": "zaid@new.com",
    "password": "1234"
}
```

200 OK • 386 ms • 319 B

```
{
    "message": "successfully retrived documents",
    "status": true,
    "documents": [
        {
            "latex": false,
            "doc_title": "test tittle",
            "id": 1
        }
    ]
```

Retrieve user-specific document:



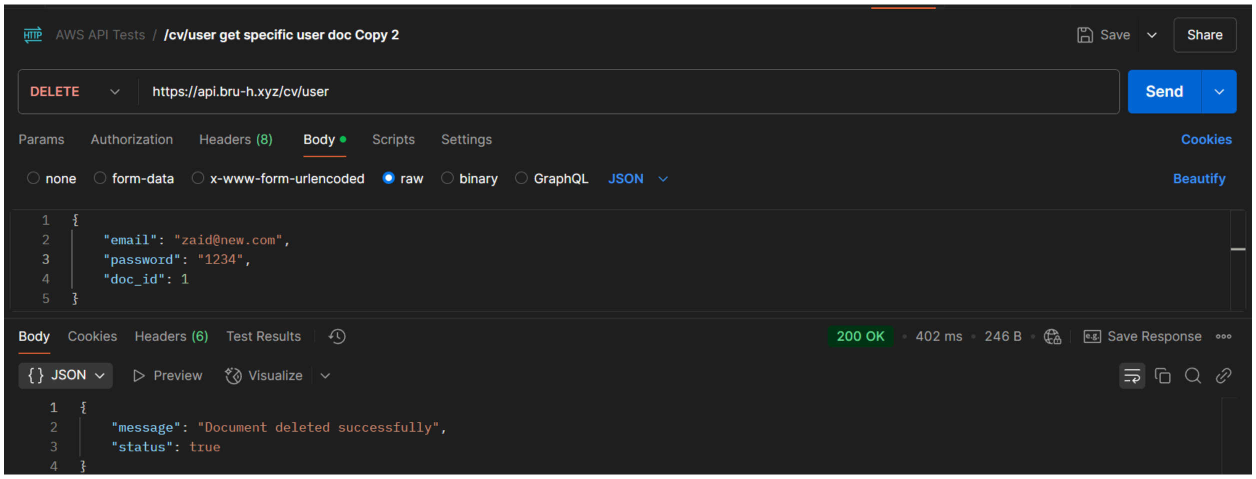
POST https://api.bru-h.xyz/cv/user

```
{
    "email": "zaid@new.com",
    "password": "1234",
    "doc_id": 1
}
```

200 OK • 387 ms • 327 B

```
{
    "message": "Document found",
    "status": true,
    "document": {
        "latex": false,
        "doc_title": "test tittle",
        "doc_body": "Doc bodyyyyy",
        "id": 1
    }
}
```

Delete user document:



AWS API Tests / /cv/user get specific user doc Copy 2

DELETE https://api.bru-h.xyz/cv/user

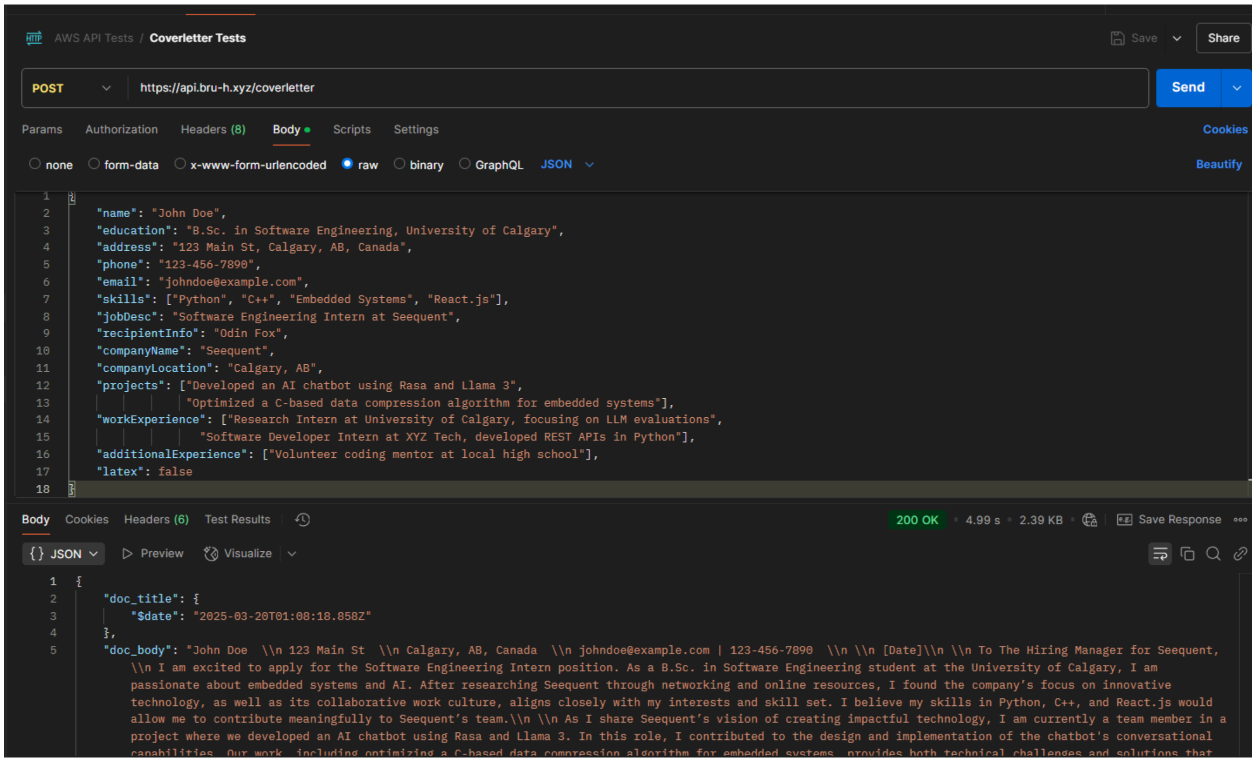
Body: raw JSON

```
{
    "email": "zaid@new.com",
    "password": "1234",
    "doc_id": 1
}
```

200 OK • 402 ms • 246 B

```
{
    "message": "Document deleted successfully",
    "status": true
}
```

Cover letter prompt test:



AWS API Tests / Coverletter Tests

POST https://api.bru-h.xyz/coverletter

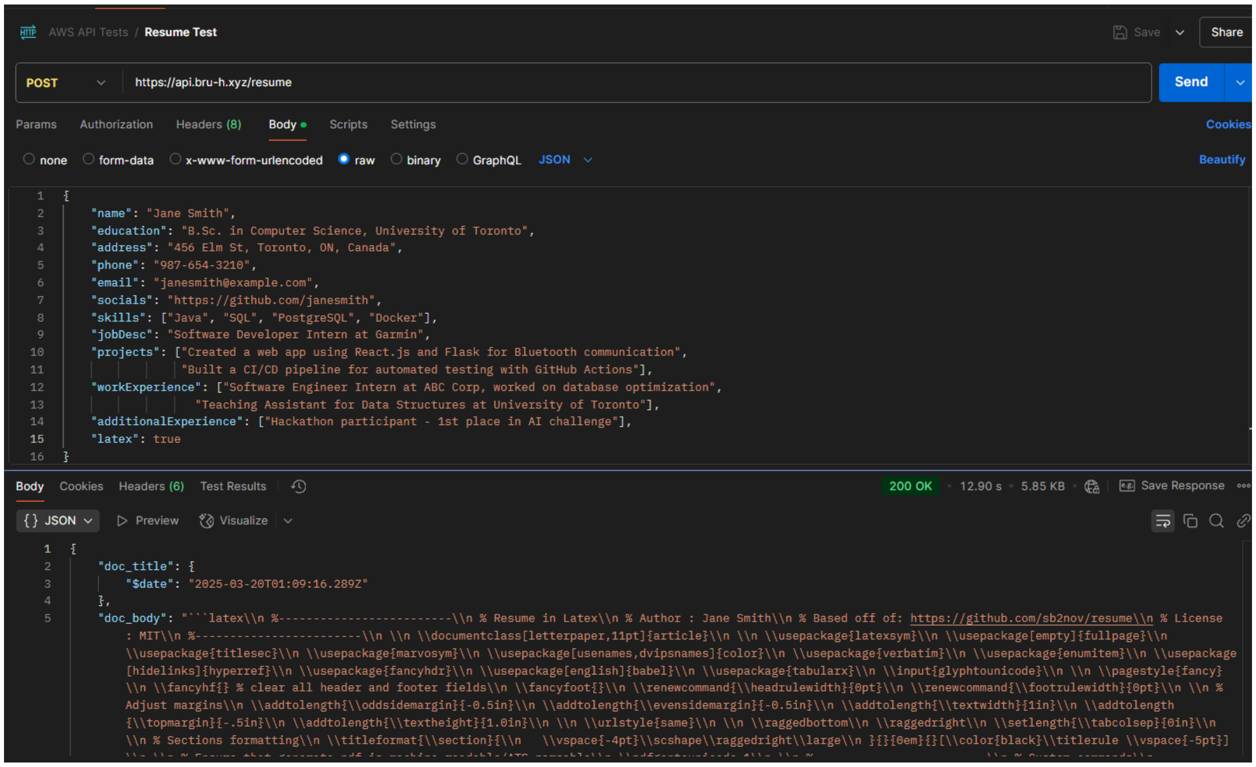
Body: raw JSON

```
{
    "name": "John Doe",
    "education": "B.Sc. in Software Engineering, University of Calgary",
    "address": "123 Main St, Calgary, AB, Canada",
    "phone": "123-456-7890",
    "email": "johndoe@example.com",
    "skills": ["Python", "C++", "Embedded Systems", "React.js"],
    "jobDesc": "Software Engineering Intern at Seequent",
    "recipientInfo": "Odin Fox",
    "companyName": "Seequent",
    "companyLocation": "Calgary, AB",
    "projects": ["Developed an AI chatbot using Rasa and Llama 3",
                 "Optimized a C-based data compression algorithm for embedded systems"],
    "workExperience": ["Research Intern at University of Calgary, focusing on LLM evaluations",
                       "Software Developer Intern at XYZ Tech, developed REST APIs in Python"],
    "additionalExperience": ["Volunteer coding mentor at local high school"],
    "latex": false
}
```

200 OK • 4.99 s • 2.39 KB

```
{
    "doc_title": {
        "$date": "2025-03-20T01:08:18.858Z"
    },
    "doc_body": "John Doe  \\n 123 Main St  \\n Calgary, AB, Canada  \\n johndoe@example.com | 123-456-7890  \\n \\n [Date]\\n \\n To The Hiring Manager for Seequent,
      \\n I am excited to apply for the Software Engineering Intern position. As a B.Sc. in Software Engineering student at the University of Calgary, I am
      passionate about embedded systems and AI. After researching Seequent through networking and online resources, I found the company's focus on innovative
      technology, as well as its collaborative work culture, aligns closely with my interests and skill set. I believe my skills in Python, C++, and React.js would
      allow me to contribute meaningfully to Seequent's team.\\n \\n As I share Seequent's vision of creating impactful technology, I am currently a team member in a
      project where we developed an AI chatbot using Rasa and Llama 3. In this role, I contributed to the design and implementation of the chatbot's conversational
      capabilities. Our work, including optimizing a C-based data compression algorithm for embedded systems, provides both technical challenges and solutions that
```

Resume prompt test:



AWS API Tests / Resume Test

POST https://api.bru-h.xyz/resume

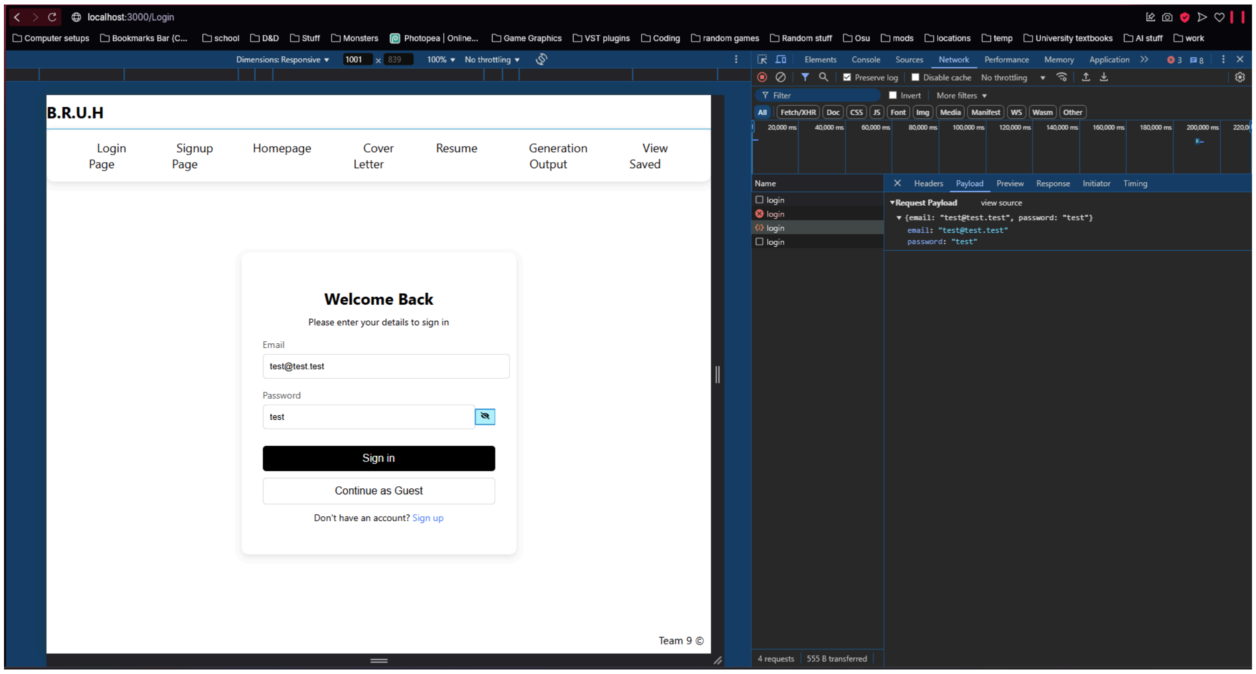
```
{
    "name": "Jane Smith",
    "education": "B.Sc. in Computer Science, University of Toronto",
    "address": "456 Elm St, Toronto, ON, Canada",
    "phone": "987-654-3210",
    "email": "janesmith@example.com",
    "socials": "https://github.com/janesmith",
    "skills": ["Java", "SQL", "PostgreSQL", "Docker"],
    "jobDesc": "Software Developer Intern at Garmin",
    "projects": ["Created a web app using React.js and Flask for Bluetooth communication",
                 "Built a CI/CD pipeline for automated testing with GitHub Actions"],
    "workExperience": ["Software Engineer Intern at ABC Corp, worked on database optimization",
                       "Teaching Assistant for Data Structures at University of Toronto"],
    "additionalExperience": ["Hackathon participant - 1st place in AI challenge"],
    "latex": true
}
```

200 OK · 12.90 s · 5.85 KB

```
{
    "doc_title": {
        "$date": "2025-03-20T01:09:16.289Z"
    },
    "doc_body": "```latex\\n %-------------------------\\n % Resume in Latex\\n % Author : Jane Smith\\n % Based off of: https://github.com/sb2nov/resume\\n % License
        : MIT\\n %------------------------\\n \\n \\documentclass[letterpaper,11pt]{article}\\n \\n \\usepackage{latexsym}\\n \\usepackage[empty]{fullpage}\\n
        \\usepackage{titlesec}\\n \\usepackage{marvosym}\\n \\usepackage[usenames,dvipsnames]{color}\\n \\usepackage{verbatim}\\n \\usepackage{enumitem}\\n \\usepackage
        [hidelinks]{hyperref}\\n \\usepackage{fancyhdr}\\n \\usepackage[english]{babel}\\n \\usepackage{tabularx}\\n \\input{glyphtounicode}\\n \\n \\pagestyle{fancy}
        \\n \\fancyhf{} % clear all header and footer fields\\n \\fancyfoot{}\\n \\renewcommand{\\headrulewidth}{0pt}\\n \\renewcommand{\\footrulewidth}{0pt}\\n \\n %
        Adjust margins\\n \\addtolength{\\oddsidemargin}{-0.5in}\\n \\addtolength{\\evensidemargin}{-0.5in}\\n \\addtolength{\\textwidth}{1in}\\n \\addtolength
        {\\topmargin}{-.5in}\\n \\addtolength{\\textheight}{1.0in}\\n \\n \\urlstyle{same}\\n \\n \\raggedbottom\\n \\raggedright\\n \\setlength{\\tabcolsep}{0in}\\n
        \\n % Sections formatting\\n \\titleformat{\\section}{\\n   \\vspace{-4pt}\\scshape\\raggedright\\large\\n }{}{0em}{}[\\color{black}\\titlerule \\vspace{-5pt}]
```

- Fetch requests were used to send user inputs from the backend to the frontend. These photos show the fetch statements that contain JSON strings being sent to the API. They act as unit tests for the functionality of the webpages.

Login Fetch Test:



localhost:3000/Login

B.R.U.H

Login Page | Signup Page | Homepage | Cover Letter | Resume | Generation Output | View Saved

Welcome Back

Please enter your details to sign in

Email: test@test.test

Password: test

Sign in

Continue as Guest

Don't have an account? Sign up

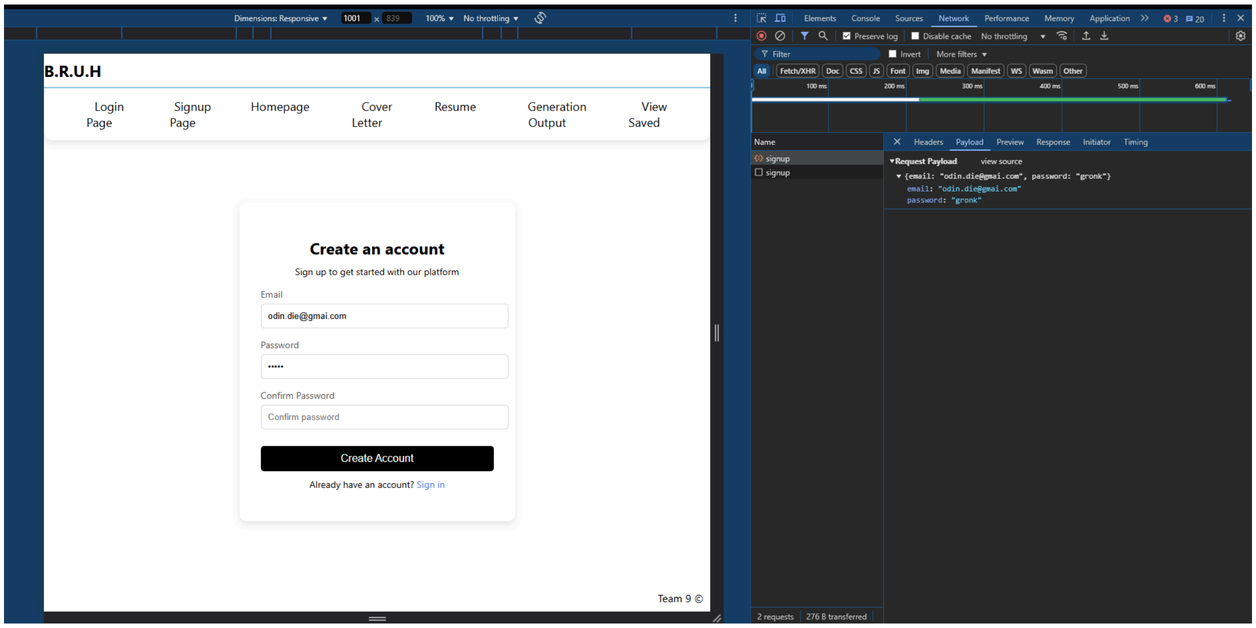
Team 9 ©

Network: login, login, login, login

```
▾Request Payload    view source
  ▾ {email: "test@test.test", password: "test"}
      email: "test@test.test"
      password: "test"
```

4 requests | 555 B transferred

Signup Fetch Test:



B.R.U.H

Login Page | Signup Page | Homepage | Cover Letter | Resume | Generation Output | View Saved

Create an account

Sign up to get started with our platform

Email: odin.die@gmai.com

Password: •••••

Confirm Password: Confirm password

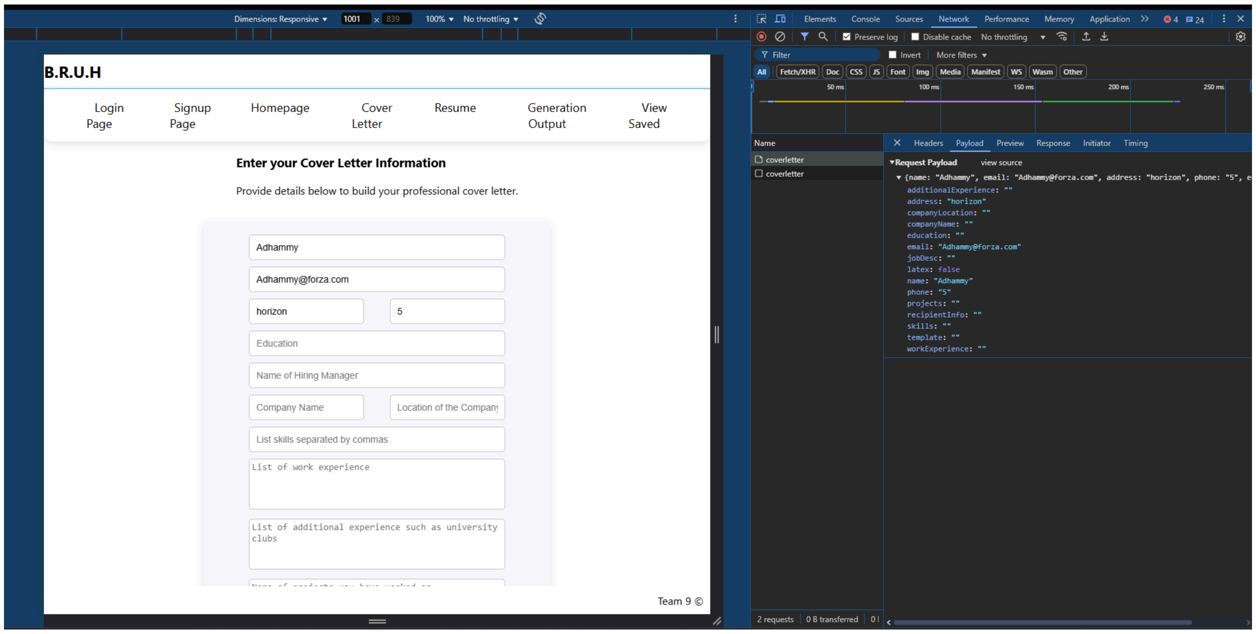
Create Account

Already have an account? Sign in

Team 9 ©

```
▾Request Payload      view source
 ▾ {email: "odin.die@gmai.com", password: "gronk"}
    email: "odin.die@gmai.com"
    password: "gronk"
```

Cover Letter Fetch Test:



B.R.U.H

Login Page | Signup Page | Homepage | Cover Letter | Resume | Generation Output | View Saved

Enter your Cover Letter Information

Provide details below to build your professional cover letter.

Adhammy

Adhammy@forza.com

horizon | 5

Education

Name of Hiring Manager

Company Name | Location of the Company

List skills separated by commas

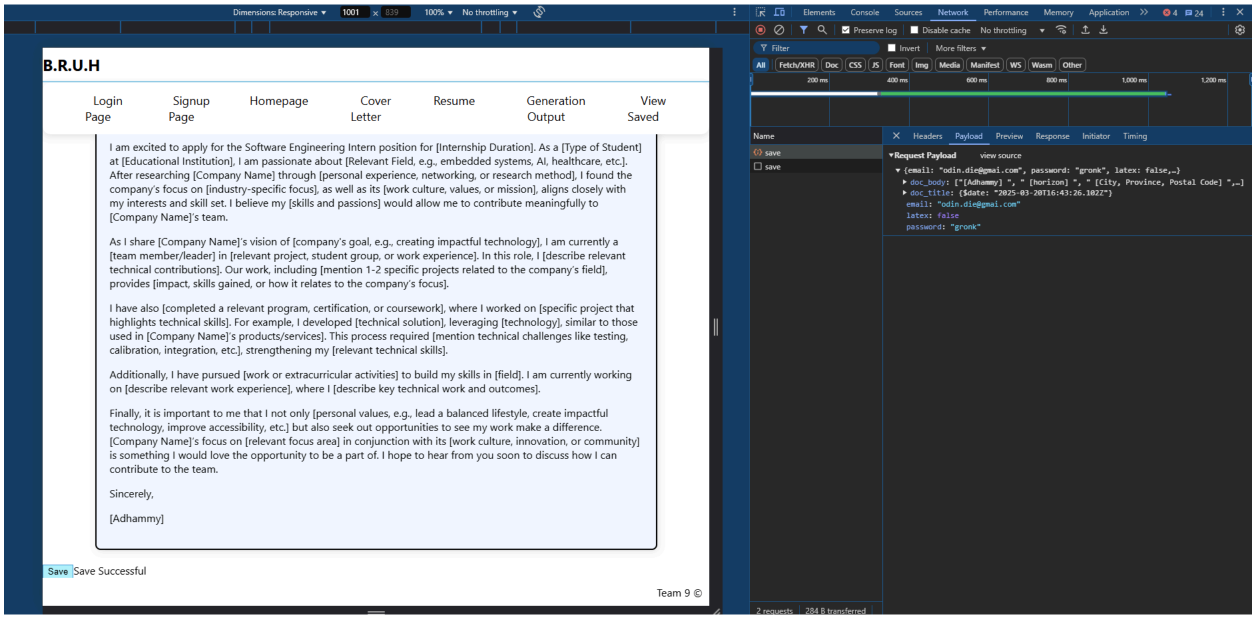
List of work experience

List of additional experience such as university clubs

Team 9 ©

```
▾Request Payload      view source
 ▾ {name: "Adhammy", email: "Adhammy@forza.com", address: "horizon", phone: "5", e
    additionalExperience: ""
    address: "horizon"
    companyLocation: ""
    companyName: ""
    education: ""
    email: "Adhammy@forza.com"
    jobDesc: ""
    latex: false
    name: "Adhammy"
    phone: "5"
    projects: ""
    recipientInfo: ""
    skills: ""
    template: ""
    workExperience: ""
```

Save Button Fetch Test:



I am excited to apply for the Software Engineering Intern position for [Internship Duration]. As a [Type of Student] at [Educational Institution], I am passionate about [Relevant Field, e.g., embedded systems, AI, healthcare, etc.]. After researching [Company Name] through [personal experience, networking, or research method], I found the company's focus on [industry-specific focus], as well as its [work culture, values, or mission], aligns closely with my interests and skill set. I believe my [skills and passions] would allow me to contribute meaningfully to [Company Name]'s team.

As I share [Company Name]'s vision of [company's goal, e.g., creating impactful technology], I am currently a [team member/leader] in [relevant project, student group, or work experience]. In this role, I [describe relevant technical contributions]. Our work, including [mention 1-2 specific projects related to the company's field], provides [impact, skills gained, or how it relates to the company's focus].

I have also [completed a relevant program, certification, or coursework], where I worked on [specific project that highlights technical skills]. For example, I developed [technical solution], leveraging [technology], similar to those used in [Company Name]'s products/services]. This process required [mention technical challenges like testing, calibration, integration, etc.], strengthening my [relevant technical skills].

Additionally, I have pursued [work or extracurricular activities] to build my skills in [field]. I am currently working on [describe relevant work experience], where I [describe key technical work and outcomes].

Finally, it is important to me that I not only [personal values, e.g., lead a balanced lifestyle, create impactful technology, improve accessibility, etc.] but also seek out opportunities to see my work make a difference. [Company Name]'s focus on [relevant focus area] in conjunction with its [work culture, innovation, or community] is something I would love the opportunity to be a part of. I hope to hear from you soon to discuss how I can contribute to the team.

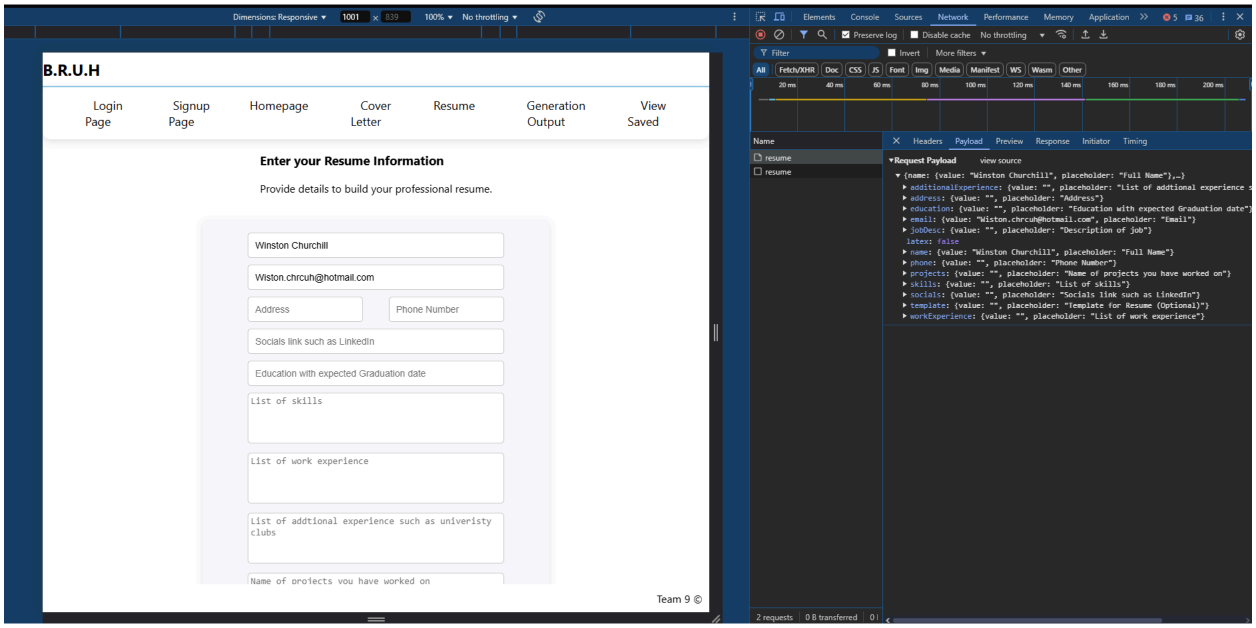
Sincerely,

[Adhammy]

Save Successful

```
▼Request Payload    view source
▼ {email: "odin.die@gmai.com", password: "gronk", latex: false,…}
  ▶ doc_body: ["[Adhammy] ", " [horizon] ", " [City, Province, Postal Code] ",…]
  ▶ doc_title: {$date: "2025-03-20T16:43:26.102Z"}
    email: "odin.die@gmai.com"
    latex: false
    password: "gronk"
```

Resume Fetch Test:



Enter your Resume Information

Provide details to build your professional resume.

Winston Churchill

Wiston.chrcuh@hotmail.com

Address | Phone Number

Socials link such as LinkedIn

Education with expected Graduation date

List of skills

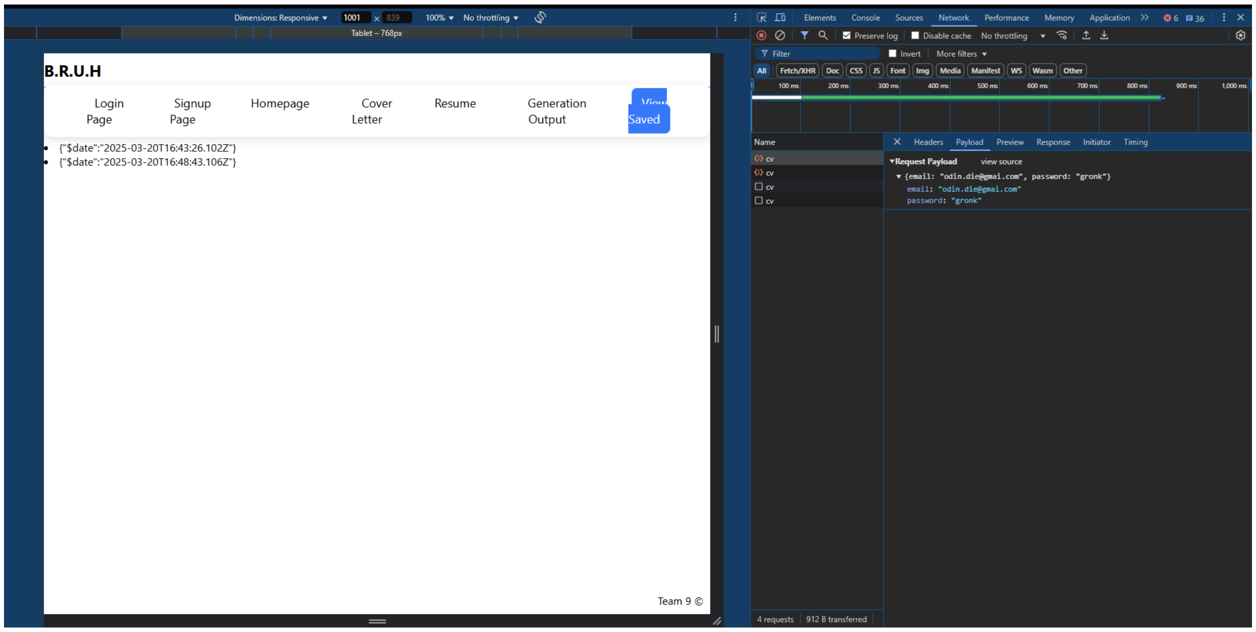
List of work experience

List of addtional experience such as univeristy clubs

Name of projects you have worked on

```
▼Request Payload    view source
▼ {name: {value: "Winston Churchill", placeholder: "Full Name"},…}
  ▶ additionalExperience: {value: "", placeholder: "List of addtional experience s
  ▶ address: {value: "", placeholder: "Address"}
  ▶ education: {value: "", placeholder: "Education with expected Graduation date"}
  ▶ email: {value: "Wiston.chrcuh@hotmail.com", placeholder: "Email"}
  ▶ jobDesc: {value: "", placeholder: "Description of job"}
    latex: false
  ▶ name: {value: "Winston Churchill", placeholder: "Full Name"}
  ▶ phone: {value: "", placeholder: "Phone Number"}
  ▶ projects: {value: "", placeholder: "Name of projects you have worked on"}
  ▶ skills: {value: "", placeholder: "List of skills"}
  ▶ socials: {value: "", placeholder: "Socials link such as LinkedIn"}
  ▶ template: {value: "", placeholder: "Template for Resume (Optional)"}
  ▶ workExperience: {value: "", placeholder: "List of work experience"}
```

Saved Documents Fetch Test:

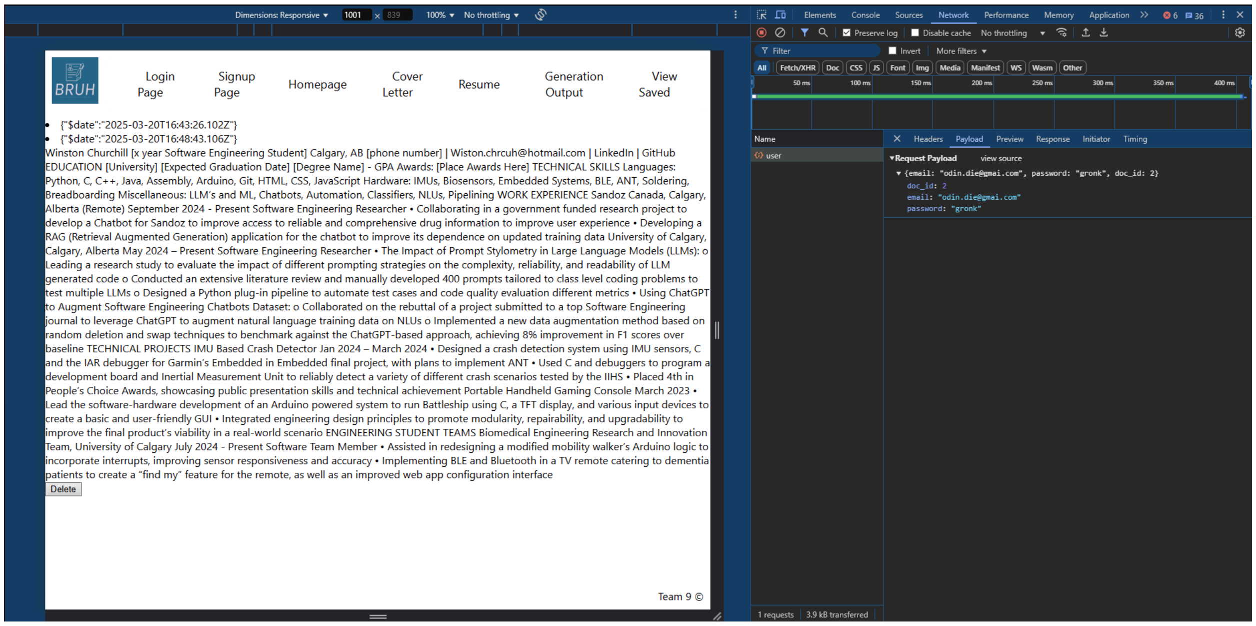


- {"$date":"2025-03-20T16:43:26.102Z"}
- {"$date":"2025-03-20T16:48:43.106Z"}

```
cv
cv
cv
cv
```

```
▼Request Payload
 ▼ {email: "odin.die@gmai.com", password: "gronk"}
    email: "odin.die@gmai.com"
    password: "gronk"
```

Grab Specific Document Test:



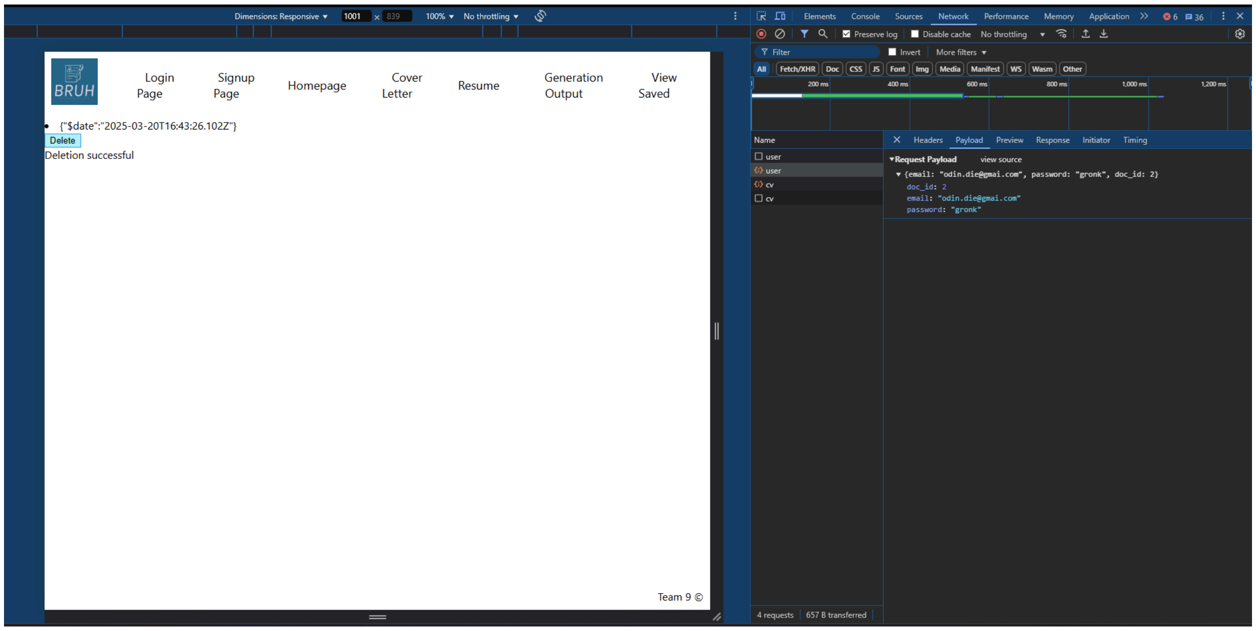
- {"$date":"2025-03-20T16:43:26.102Z"}
- {"$date":"2025-03-20T16:48:43.106Z"}

Winston Churchill [x year Software Engineering Student] Calgary, AB [phone number] | Wiston.chrcuh@hotmail.com | LinkedIn | GitHub EDUCATION [University] [Expected Graduation Date] [Degree Name] - GPA Awards: [Place Awards Here] TECHNICAL SKILLS Languages: Python, C, C++, Java, Assembly, Arduino, Git, HTML, CSS, JavaScript Hardware: IMUs, Biosensors, Embedded Systems, BLE, ANT, Soldering, Breadboarding Miscellaneous: LLM's and ML, Chatbots, Automation, Classifiers, NLUs, Pipelining WORK EXPERIENCE Sandoz Canada, Calgary, Alberta (Remote) September 2024 - Present Software Engineering Researcher • Collaborating in a government funded research project to develop a Chatbot for Sandoz to improve access to reliable and comprehensive drug information to improve user experience • Developing a RAG (Retrieval Augmented Generation) application for the chatbot to improve its dependence on updated training data University of Calgary, Calgary, Alberta May 2024 – Present Software Engineering Researcher • The Impact of Prompt Stylometry in Large Language Models (LLMs): o Leading a research study to evaluate the impact of different prompting strategies on the complexity, reliability, and readability of LLM generated code o Conducted an extensive literature review and manually developed 400 prompts tailored to class level coding problems to test multiple LLMs o Designed a Python plug-in pipeline to automate test cases and code quality evaluation different metrics • Using ChatGPT to Augment Software Engineering Chatbots Dataset: o Collaborated on the rebuttal of a project submitted to a top Software Engineering journal to leverage ChatGPT to augment natural language training data on NLUs o Implemented a new data augmentation method based on random deletion and swap techniques to benchmark against the ChatGPT-based approach, achieving 8% improvement in F1 scores over baseline TECHNICAL PROJECTS IMU Based Crash Detector Jan 2024 – March 2024 • Designed a crash detection system using IMU sensors, C and the IAR debugger for Garmin's Embedded in Embedded final project, with plans to implement ANT • Used C and debuggers to program a development board and Inertial Measurement Unit to reliably detect a variety of different crash scenarios tested by the IIHS • Placed 4th in People's Choice Awards, showcasing public presentation skills and technical achievement Portable Handheld Gaming Console March 2023 • Lead the software-hardware development of an Arduino powered system to run Battleship using C, a TFT display, and various input devices to create a basic and user-friendly GUI • Integrated engineering design principles to promote modularity, repairability, and upgradability to improve the final product's viability in a real-world scenario ENGINEERING STUDENT TEAMS Biomedical Engineering Research and Innovation Team, University of Calgary July 2024 - Present Software Team Member • Assisted in redesigning a modified mobility walker's Arduino logic to incorporate interrupts, improving sensor responsiveness and accuracy • Implementing BLE and Bluetooth in a TV remote catering to dementia patients to create a "find my" feature for the remote, as well as an improved web app configuration interface

```
user
```

```
▼Request Payload
 ▼ {email: "odin.die@gmai.com", password: "gronk", doc_id: 2}
    doc_id: 2
    email: "odin.die@gmai.com"
    password: "gronk"
```

Delete Document Test:



- {"$date":"2025-03-20T16:43:26.102Z"}

Deletion successful

```
Name
user
user
cv
cv
```

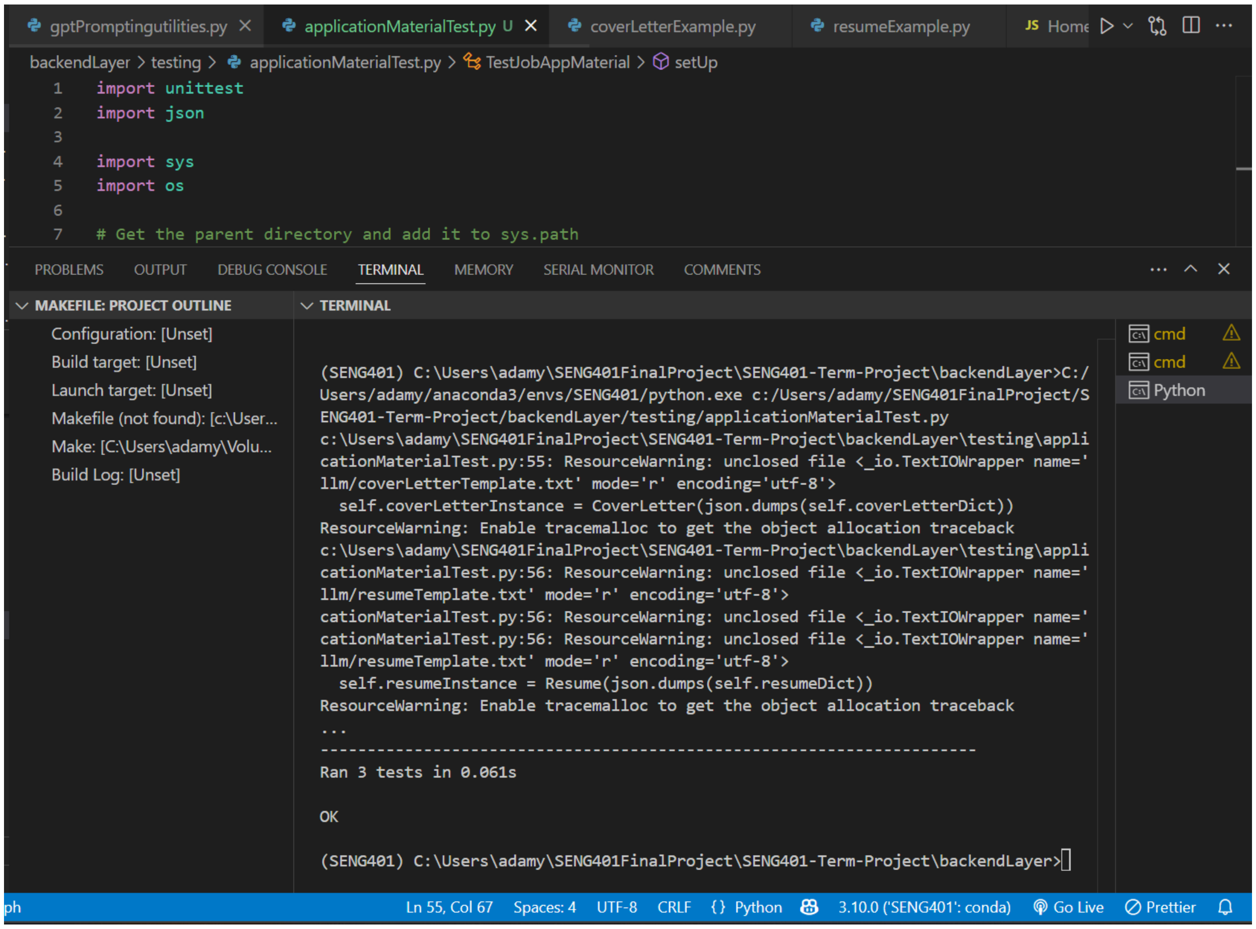
```
▾Request Payload    view source
  ▾ {email: "odin.die@gmai.com", password: "gronk", doc_id: 2}
      doc_id: 2
      email: "odin.die@gmai.com"
      password: "gronk"
```

Unit testing:

- One of the most important parts of our application is the application material file and its associated objects. This class is important in determining the information each resume/cover letter should have and constructing the prompt needed to create them. As a result, creating unit tests for this particular class was important.

Application material test cases:

```
import unittest
import json

import sys
import os

# Get the parent directory and add it to sys.path
parent_dir = os.path.abspath(os.path.join(os.path.dirname(__file__), "../"))
sys.path.append(parent_dir)

from applicationMaterial import (Resume, CoverLetter)

class TestJobAppMaterial(unittest.TestCase):

  def setUp(self):
    self.resumeDict = {
      "name": "Jane Smith",
      "education": "B.Sc. in Computer Science, University of Toronto",
      "address": "456 Elm St, Toronto, ON, Canada",
      "phone": "987-654-3210",
      "email": "janesmith@example.com",
      "socials": "https://github.com/janesmith",
      "skills": ["Java", "SQL", "PostgreSQL", "Docker"],
      "jobDesc": "Software Developer Intern at Garmin",
      "projects": ["Created a web app using React.js and Flask for Bluetooth
communication",
                "Built a CI/CD pipeline for automated testing with GitHub Actions"],
      "workExperience": ["Software Engineer Intern at ABC Corp, worked on database
optimization",
                "Teaching Assistant for Data Structures at University of Toronto"],
      "additionalExperience": ["Hackathon participant - 1st place in AI challenge"],
      "template": "",
      "latex": "False"
    }

    self.coverLetterDict = {
      "name": "John Doe",
      "education": "B.Sc. in Software Engineering, University of Calgary",
      "address": "123 Main St, Calgary, AB, Canada",
      "phone": "123-456-7890",
      "email": "johndoe@example.com",
      "skills": ["Python", "C++", "Embedded Systems", "React.js"],
      "jobDesc": "Software Engineering Intern at Seequent",
      "recipientInfo": "Odin Fox",
      "companyName": "Seequent",
      "companyLocation": "Calgary, AB",
      "projects": ["Developed an AI chatbot using Rasa and Llama 3",
                "Optimized a C-based data compression algorithm for embedded systems"],
      "workExperience": ["Research Intern at University of Calgary, focusing on LLM
evaluations",
                "Software Developer Intern at XYZ Tech, developed REST APIs in Python"],
      "additionalExperience": ["Volunteer coding mentor at local high school"],
      "template": "",
      "latex": "False"
    }

    # Convert to JSON strings before passing to class constructors
    self.coverLetterInstance = CoverLetter(json.dumps(self.coverLetterDict))
    self.resumeInstance = Resume(json.dumps(self.resumeDict))

  def test_attributes_initialization(self):
    """Test if attributes are correctly initialized from JSON input"""
    self.assertEqual(self.coverLetterInstance.name, "John Doe")
    self.assertEqual(self.coverLetterInstance.education, "B.Sc. in Software Engineering,
University of Calgary")
    self.assertEqual(self.coverLetterInstance.skills, ["Python", "C++", "Embedded Systems",
"React.js"])
    self.assertEqual(self.coverLetterInstance.materialType, "cover letter")

    self.assertEqual(self.resumeInstance.name, "Jane Smith")
    self.assertEqual(self.resumeInstance.education, "B.Sc. in Computer Science, University
of Toronto")
      self.assertEqual(self.resumeInstance.skills, ["Java", "SQL", "PostgreSQL", "Docker"])
      self.assertEqual(self.resumeInstance.materialType, "resume")

  def test_create_cover_letter_prompt(self):
      """Test if createCoverLetterPrompt() generates expected prompt content"""
      prompt = self.coverLetterInstance.createCoverLetterPrompt()
      self.assertIn("John Doe", prompt)
      self.assertIn("Software Engineering Intern", prompt)
      self.assertIn("Seequent", prompt)
      self.assertIn("B.Sc. in Software Engineering", prompt)
      self.assertIn("Python", prompt)

  def test_create_resume_prompt(self):
      """Test if createResumeLetterPrompt() generates expected prompt content"""
      prompt = self.resumeInstance.createResumeLetterPrompt()
      self.assertIn("Jane Smith", prompt)
      self.assertIn("Software Developer Intern", prompt)
      self.assertIn("University of Toronto", prompt)
      self.assertIn("Java", prompt)
      self.assertIn("https://github.com/janesmith", prompt)

if __name__ == '__main__':
  unittest.main()
```



```python
import unittest
import json

import sys
import os

# Get the parent directory and add it to sys.path
```

MAKEFILE: PROJECT OUTLINE

Configuration: [Unset]
Build target: [Unset]
Launch target: [Unset]
Makefile (not found): [c:\User...
Make: [C:\Users\adamy\Volu...
Build Log: [Unset]

TERMINAL

```
(SENG401) C:\Users\adamy\SENG401FinalProject\SENG401-Term-Project\backendLayer>C:/
Users/adamy/anaconda3/envs/SENG401/python.exe c:/Users/adamy/SENG401FinalProject/S
ENG401-Term-Project/backendLayer/testing/applicationMaterialTest.py
c:\Users\adamy\SENG401FinalProject\SENG401-Term-Project\backendLayer\testing\appli
cationMaterialTest.py:55: ResourceWarning: unclosed file <_io.TextIOWrapper name='
llm/coverLetterTemplate.txt' mode='r' encoding='utf-8'>
  self.coverLetterInstance = CoverLetter(json.dumps(self.coverLetterDict))
ResourceWarning: Enable tracemalloc to get the object allocation traceback
c:\Users\adamy\SENG401FinalProject\SENG401-Term-Project\backendLayer\testing\appli
cationMaterialTest.py:56: ResourceWarning: unclosed file <_io.TextIOWrapper name='
llm/resumeTemplate.txt' mode='r' encoding='utf-8'>
cationMaterialTest.py:56: ResourceWarning: unclosed file <_io.TextIOWrapper name='
cationMaterialTest.py:56: ResourceWarning: unclosed file <_io.TextIOWrapper name='
llm/resumeTemplate.txt' mode='r' encoding='utf-8'>
  self.resumeInstance = Resume(json.dumps(self.resumeDict))
ResourceWarning: Enable tracemalloc to get the object allocation traceback
...
----------------------------------------------------------------------
Ran 3 tests in 0.061s

OK

(SENG401) C:\Users\adamy\SENG401FinalProject\SENG401-Term-Project\backendLayer>
```