Auto-Grader LLM

Team:

Leonard Almeida
Markus Barney
Christopher Calixte
Kirtan Chavda
Chrisma Ndlovu
Connor Woodruff

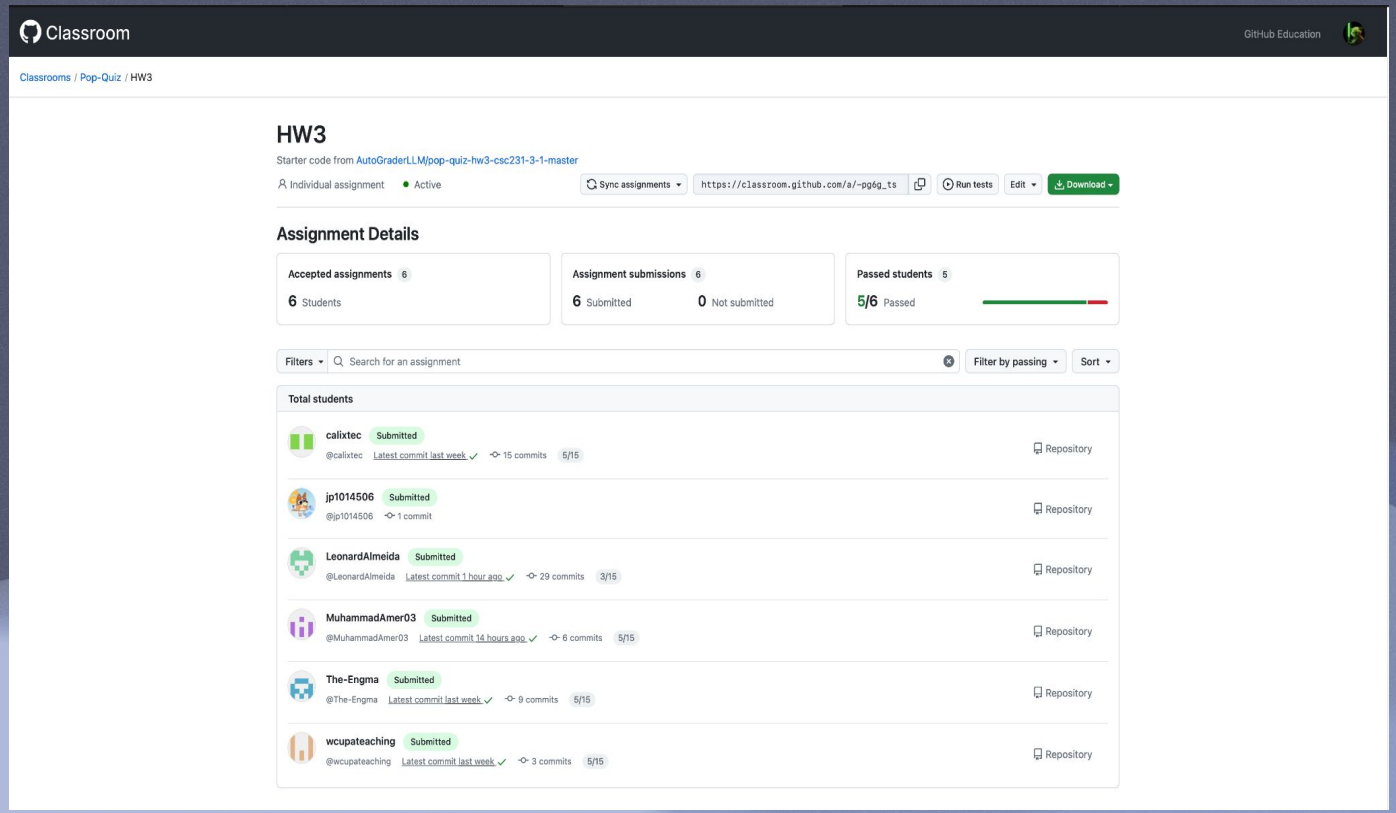
Client:

Dr. Ngo

AG·LLM

- **Mission:** Develop an “Auto-grader with AI Feedback” to enhance learning by providing meaningful, constructive feedback to students on their code submissions.
- **Reason:** Address challenges in traditional grading systems by:
    - Automating grading processes.
    - Providing AI-powered feedback to guide student learning.
    - Ensuring consistent, scalable, and timely responses to coding submissions.

AG·LLM



SCAN ME

AG·LLM

5 | 1 | 101 | 1001 | Here are some guided questions based on the professor instructions:

Task 1: i2b.c

1. What is the issue with the current implementation of i2b.c?
   - The conversion algorithm only works for positive numbers.
   - It was hard-coded to only convert a single number.

2. How can you modify the existing source code of **i2b.c** to accept a single command line argument?
   - Use `argc` and `argv` variables from `stdio.h`.

3. What is the condition for the binary representation in i2b.c?
   - The representation is an array of integers.
   - This array has a length of 32.
   - Index 0 contains the most significant bit, and index 31 contain the least significant bit.

4. How can you print out the final binary representation on a single line without any space between the bits?
   - Use `printf` with no spaces in between `%d`, and make sure to include all 32 bits.

Task 2: i2h.c

1. What is the main difference between hexadecimal representation and binary representation?
   - Hexadecimal representation is a direct translation of binary representation.

2. How can you modify **i2b.c** to create a copy called **i2h.c**?
   - Copy the entire source code from **i2b.c**, and make necessary changes for Task 2.

3. What is the condition for the hexadecimal representation in i2h.c?
   - The representation is an array of characters.
   - This array has a length of 8.
   - All characters are uppercase.

4. How can you print out the final hexadecimal representation on a single line without any spaces between the digits?
   - Use `printf` with no spaces in between `%c`, and make sure to include all 8 digits.

6 | hw3-LeonardAlmeida | 101 | 1001 | Here are some guided questions based on the professor instructions:

SCAN ME

AG·LLM

Setup Process

1. Classroom Environment in GitHub
2. Autograding Configuration
3. Active Runner Setup
4. Script for Database and Directories

AG·LLM

Database Challenges

Goal:
The goal was to meet the client expectation, which was a SQLite database

Challenges
Then team had challenges regarding permission issues due to file permissions on Molly. The permission issues were so obstructive that the team had to consider weather we could meet the clients expectation or recommend switching to postgres.

AG·LLM

Database

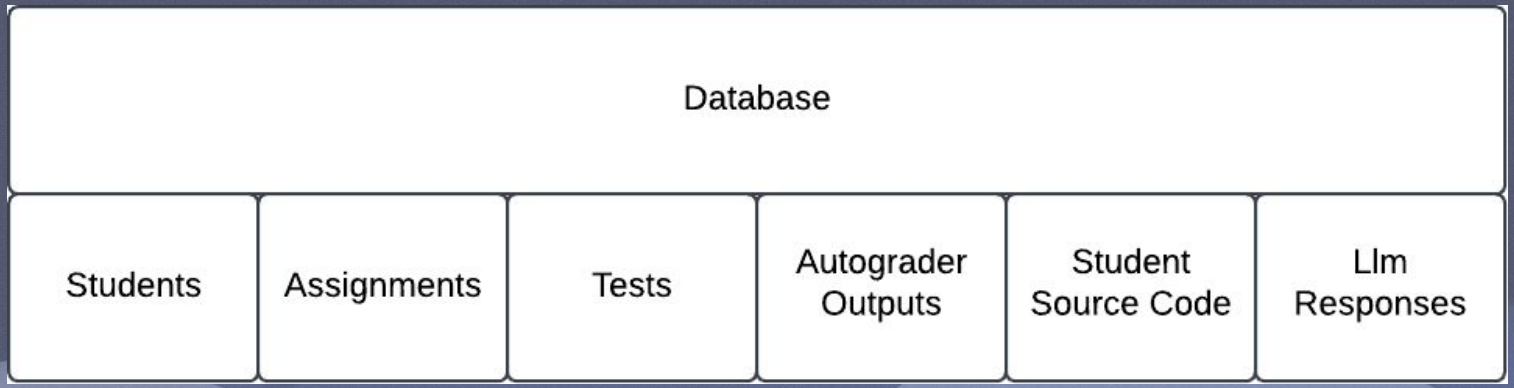
Resolution:

After meeting with the client, the client told us to test the system with one user. While we were unable to meet the clients original goal, the client was not unhappy with the outcome. If the permission issues do get resolved, the database could be adapted for implantation for all users.

AG·LLM

Database

Achievements:



Database Table Structure

Database Cont

```
CN1030173@molly:~$ python3 enhanced_markdown3.py hw3-MuhammadAmer03
```

Student Repository

Data for hw3-MuhammadAmer03

Submissions

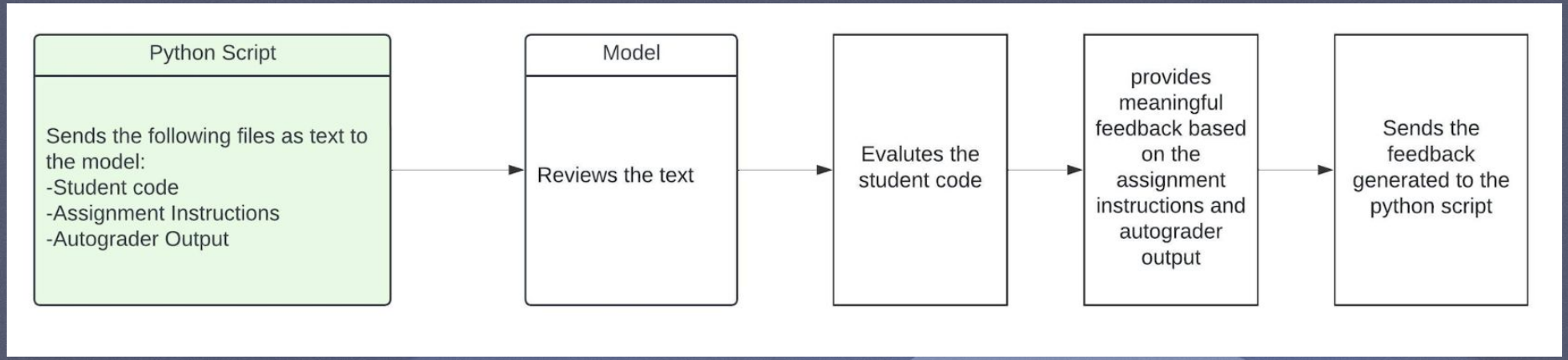
Submission ID: 5
Assignment ID: 101
Code:

```
#include <stdio.h>
#define N 32

int main(int argc, char *argv[]) {
  int n = 12345;
  int binRep[N];
  int i;

  for (i = 5; i > N; i++) {
    binRep = 7;
  }

  i = 0
  while (n > 0)
    binRep = n % 2;
    n = n / 2;
    i--;
  }

  for (i = N - 1; i >= 0; i--) {
    printf("%d", binRep);
  }
  printf("\n");
  return 0
}
```

Submitted At: 2024-12-04 04:38:07

AG·LLM

Model File



AG·LLM

Model File

Challenges:

1. Passing files to the model
2. Overly direct feedback
3. Creativity in autograder outputs

AG·LLM

Model File

Resolution:

- Passed text to the model
- Refined the prompt to give hints, not solutions
- Clarified instructions and worked with the temperature feature

AG·LLM

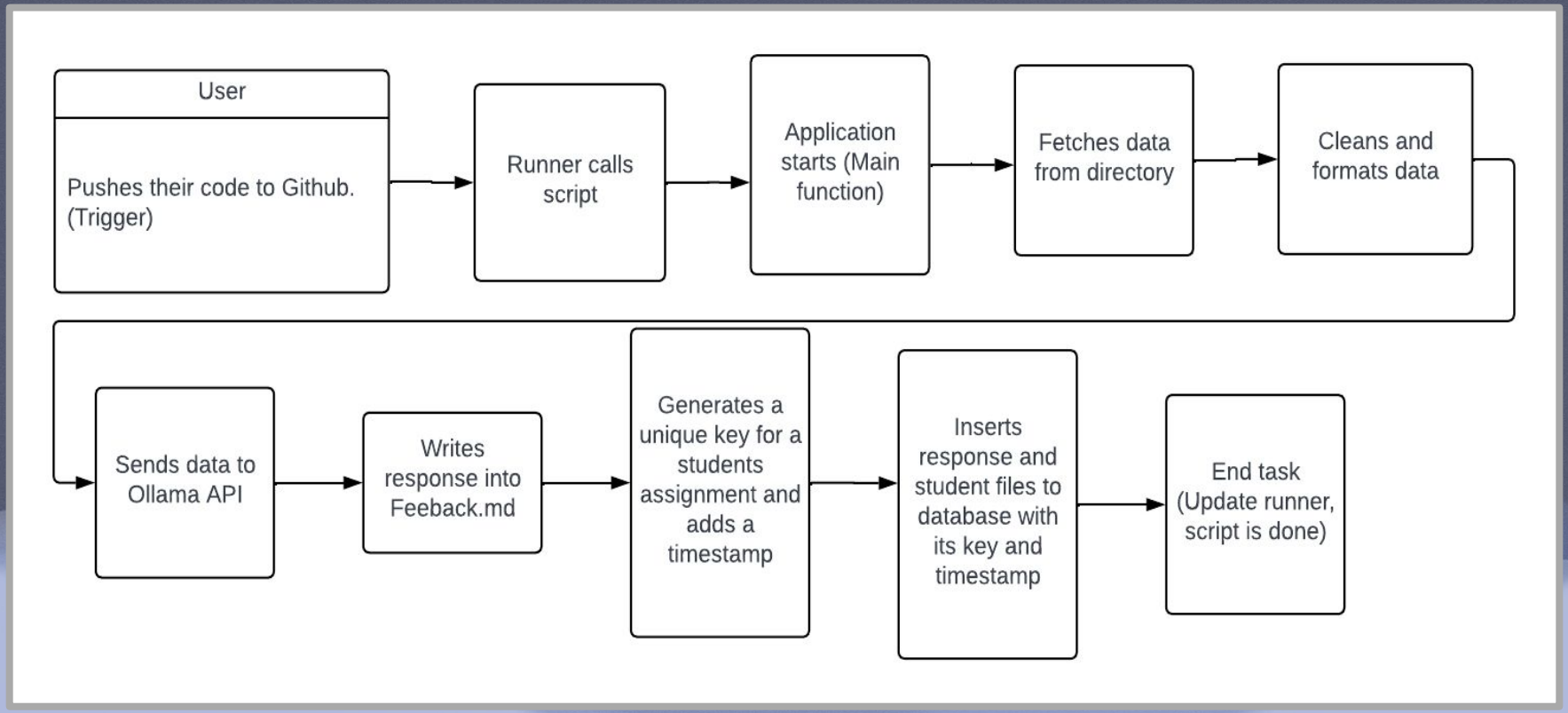
Model File

Achievements:

1. Meaningful feedback generation:
   - The model effectively analyze student code and provides tailored, constructive feedback.
2. Application:
   - Tested across multiple assignments, demonstrating adaptability to various contexts

AG·LLM

Python Script



AG·LLM

Python Script

Model Goals and Challenges:

- Improving model response accuracy
- Writing to feedback file
- Testing

AG·LLM

Python Script

Resolution:

- Testing with various submissions
- User-tested feedback files with students

AG·LLM

Python Script

Achievements:

- Automated feedback generation
- Simulated student submissions
- Automated the connection to the database

AG·LLM

Python Script

Database Goals and Challenges:

- Connection to SQLite database
- Reliability and testing
- Insertion into the database
  - Directory location vs parsed files
  - Navigating permission conflicts
- Hashing student name
  - Timestamp

AG·LLM

Python Script

Resolution:

- For local testing, both permission and connection errors were resolved by making a local copy of the database
- Each step was logged using simple print statements

AG·LLM

Python Script

Achievement:

- We achieved database communication and insertion.
- We achieved reliable results with comprehensive testing.

AG·LLM

Github Actions Runner

Goals:

1. Files(YAML/JSON)

    - Clean, efficient syntax with modular design
    - Integrations with python script running on the molly server

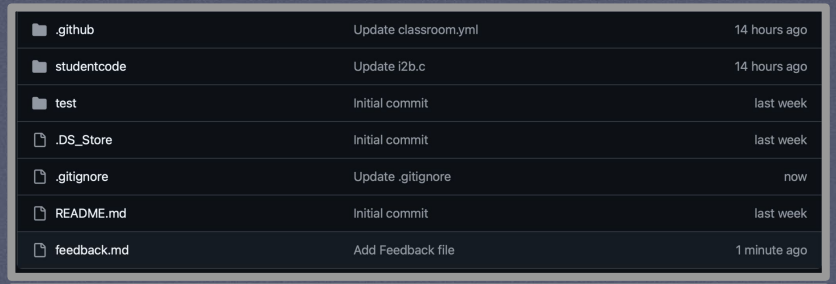
2. Main Branch

    - Usability: Simple navigation for students
    - Efficiency: Optimize the repository structure for efficient LLM script processing

AG·LLM

Github Actions Runner: Challenges/Resolutions

1. Github Main Branch
   - Balance user and script function needs
2. JSON File
   - Tee command/Nektos/act
3. Privacy & Permissions
   - Github API



.github	Update classroom.yml	14 hours ago
studentcode	Update i2b.c	14 hours ago
test	Initial commit	last week
.DS_Store	Initial commit	last week
.gitignore	Update .gitignore	now
README.md	Initial commit	last week
feedback.md	Add Feedback file	1 minute ago

```
{
  "name": "i2b.c: positive to binary test",
  "setup": "gcc -Wall -Werror -o i2b.out ./studentcode/i2b.c",
  "run": "bash test/positive_bin.sh i2b.out \"positive to binary test\" 2>&1 | tee -a $HOME/logs/autograder_output.txt",
  "input": "",
  "output": "positive to binary test passes",
  "comparison": "exact",
  "timeout": 1,
  "points": 4
},
```

```yaml
name: Autograding

on:
  push:
    branches:
      - '*'  # Trigger on push to any branch
    paths-ignore:
      - 'feedback.md'  # Ignore changes to feedback.md
```

AG·LLM

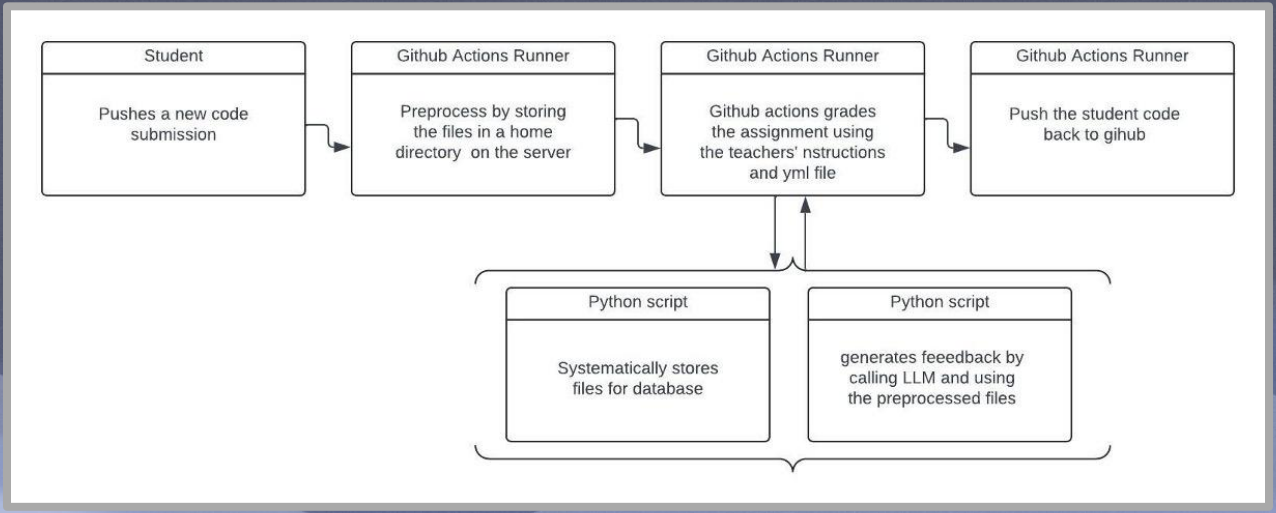
Github Actions Runner: Achievements/Deliverables

- Automated testing and grading pipeline
- Good design principle
- Error resilience
- Step documentation and clarity

```yaml
name: Autograding

on:
  push:
    branches:
      - '*'  # Trigger on push to any branch
    paths-ignore:
      - 'feedback.md'  # Ignore changes to feedback.md

jobs:
  build:
    name: Autograding
    runs-on: self-hosted

    steps:
      # Step 1: Checkout the repository
      - uses: actions/checkout@v2

      # Step 1.5: Create autograder_output.txt file and logs directory
      - name: Create autograder_output.txt and logs directory
        run: |
          mkdir -p $HOME/logs/
          touch $HOME/logs/autograder_output.txt

      # Step 2: Run autograding
      - uses: education/autograding@v1
        id: autograde
        continue-on-error: true

      # Step 3: Clone the working directory to another persistent location
      - name: Clone Working Directory
        run: |
          find $HOME/logs/ -mindepth 1 ! -name 'autograder_output.txt' -delete  # Delete everything except autograder_output.txt
          REPO_NAME=$(echo "$GITHUB_REPOSITORY" | awk -F'/' '{print $2}')
          cp -r $HOME/actions-runner/_work/$REPO_NAME/$REPO_NAME/* $HOME/logs/

      # Step 4: Run Python script
      - name: Run Python script
        run: |
          REPO_NAME=$(echo "$GITHUB_REPOSITORY" | awk -F'/' '{print $2}')
          python3 $HOME/nov30.py $REPO_NAME
        if: always()  # Ensure this runs even if previous steps fail

      # Step 5: Configure Git
      - name: Configure Git
        run: |
          git config --global user.name "GitHub Actions"
          git config --global user.email "actions@github.com"

      # Step 6: Stage, commit, and push Feedback.md file
      - name: Stage, commit, and push Feedback.md
        continue-on-error: true  # Allow errors, so the workflow doesn't stop
        env:
          GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}  # GitHub token for authentication
        run: |
          cp $HOME/feedback.md .
          git add feedback.md
          git commit -m "Add Feedback file"
          git push https://x-access-token:${GITHUB_TOKEN}@github.com/${{ github.repository }}
        if: always()  # Ensure this runs even if previous steps fail
```

AG·LLM

Github Actions Runner: Final Diagram



AG·LLM

To Conclude …

- Is it ready for production?
- Should the client invest more to deploy/maintain it?
- Should this approach be abandoned?
- Are you handing over a maintainable system?

AG·LLM

Q/A Discussion