Background:
One of our friends mentioned that the new google cloud platform was very easy to use and had some very powerful tools that could be fun to mess around with. We took a look at GCP's offerings and were very excited by what we found. We decided that we wanted to use some of the API's like the speech to text and NLP API's to build a project for Cruzhacks.

Aim:
Amnis aims to create a live lecture client for students and professors to have a more interactive and engaging learning experience. By leveraging the Google Cloud Platform's powerful tools, we hope to make online webcasts a much more interactive and relevant experience to each viewer.

Features:
The program has 2 standout features. The first being a discussion/chat section where students can ask questions regarding the lecture no matter where they are. The chat runs on a Node.js server as well as a MongoDB database to store previous questions and chat messages. The second feature is live tags. Live tags are generated using Google's speech to text and NLP API. From the professor's live mic the audio is analyzed by the API to find keywords. Those words are then put through a database and sorted based on frequency. The words are then transcribed into a text file and displayed as buttons. These buttons can be clicked and it will take the student to a Wikipedia link with additional info about the tag.

Future Plans:
We hope to bring Amnis to classes throughout our campus. We plan on bringing elements from current webcasts classes use onto Amnis so that switching to Amnis can be a nice and easy transition. In addition to adding features from other similar services, we plan on adding an upvote button for comments, so that professors can view and address questions that students are most confused about.

Technologies Used:
- Node.js
- MongoDB
- Javascript
- Websockets
- Jquery(AJAX)
- HTML
- CSS
- Python
- Google Cloud Platform Speech to Text API
- Google Cloud Platform NLP API

Installation/Dependencies:
*** In order to run the program you must have**
- Sockets.io
- Node.js
- MongoDB
- The Google API's
- Pyaudio

To run the program, you must first setup a mongo database on your local computer. Then start the web application using “node server”. Once the server is up, you can run mic.py then open the mainpage.html file. The application is currently set up to listen to input from the user's microphone and update the buttons on the page with keywords from the user's speech.