Oct 15, 2025

- Update: Front end and back end are good-but are currently hardcoded
    - Will look into how to not hardcode it
- Go over presentation slides and finalize roles

TODO:

- Need back up API keys in case we hit ‘rate limited exceeded’
    - Rui: 185310d8-df47-4c64-a536-0d24592e283c
- Add photo and name to presentation
- Look over lecture content to be prepared to use the lecturing points as reflections/takeaways
    -

Oct 14, 2025

- App: function vs Website: content
- Front-End team met on Wednesday
    - Plan to finish front end work by Tuesday night
    - push to Github Wednesday
- Conway’s Law we did- yay
- We didnt talk to stakeholder - oops its ok YOLO
- We need to push to repo throughout our week long process to show we are not pushing the final project on one solo push as that can be perceived as we are using AI to the code for us
- Figure out what are the risky parts?
    - Consider the timeline for when the product is due
    - Talk to stakeholders
    - If you are not familiar with a specific programming concept (languages, frameworks, LLM, etc)
- Think about expandability
    - Example: what if a new set of emojis become available next week, how is our project going to reflect that? Or more so how can we expand

Questions asked from both our group and other groups:

- LLM vs Direct Emoji Replacement?
    - LLM
        - What if the networks go down?
        - Maybe timeouts, fallback mechanism
        - Will it be stable?
    - Direct emoji replacement
        - Can start with this first
        - Clunky
    - Maybe do hybrid approach - fall back when LLM is done
    - What if you use an LLM to create a giant table to do lookup (i.e. a 1000 common phrases) and a giant table
- Format / what to include for presentations?

    - Show software and what you accomplished.
    - Can include reflection on challenges and successes
    - Prof wants to hear what we learned! And the process.
- Deployment?
    - Don't do GitHub pages: not meant for delivery, just a small app
        - If pressed for time, you can do this though.
    - Prof recommends maybe: Cloudflare, digital ocean, AWS
- Input/output concerns - max number of emojis?
    - Assume the worst - what if the user inputs a 1 million emojis?
    - Don't trust inputs or outputs on the internet!
    - Need to set constraints - should read the input to make sure it's not too long or has bad data in it.
- UI simplicity
    - Should be simple enough that anyone can use it!
- Google Drive
    - Don't have a google drive - MOVE EVERYTHING TO THE REPO!!
- Don't use
    - If HTML has something, use that INSTEAD of . There's probably an html tag for that.
        - I.e. <header>, <footer>, <nav>, <main>, headings (<h1>, <h2>)
    - We can fix it by using attributes and role, or semantic markup
- Creating a dropdown for Gen Z to Gen Alpha, Gen Z to Boomer?
    - Prof said don't do this - putting people into buckets

Prompt:

Oct 12, 2025

Project Manager: Abhishek

TODO:

- Identify which languages we will use
    - Front end: javascript
- For this project since it too simple → we can have some members focusing on the presentation structure also

Agenda:

- Abhiskek: showed prototype
- Audria: showed ideas for front end
- The team added to 'What are the things we need to think before we start coding?" section below

Notes:

- Abhishek made a prototype front and back end
- Rui Sun: Can we just set up a large language models on backend with prompts to translate the emojis?
- Khang: Edge cases
- Bobby:

- Also, how do we structure GitHub?
- Should we have some maintainer for the codebase?
- Like we have backend lead and front end lead who are the one can approve a PR to main?
- Or we just push
- We can ask the TA for advise
- Abhiskek's suggestion: we can have one person

What are the things we need to think before we start coding:

1. Understanding who are our users : Kids, Adults, elderly people? How do we make sure we have the least number of clicks possible per request?
    a. Who's the end user? How they want to use our software in their business or daily life
    b. Adult for professor to translate
    TODO in future: talk to stakeholder

2. How can we handle error or fault tolerance level of our software:
    a. Set an input length limitation to avoid overlong input
    b. Only allow plain text with emoji as input
3. Understanding failure cases ? like : <text> <100 emoji> in this case output make any sense be it with statistical method or using ML (we can think more of such edge cases and try to see how we can resolve them or handle them if such situation occur) (even if he don't ask about all the cases we will still have something to share like ' we kept this in consideration or that')
    a. Ideally each emoji carries its own meaning so until and unless someone decides to entirely communicate in emojis (eg. "I love coding while drinking coffee and listening to music" -> 👩‍💻❤️💻☕🎧🎶😊)
        i. For such cases we have two choices :
            1. Just return the answer considering the first emoji only and ignoring rest.
            2. Using AI as backup option (But for that first step is identifying when our basic approach fails as there could be 100s failure cases)
4. How to handle context ambiguity (one emoji can have different meanings under different contexts)
    a. If we use LLM for translation, we can ask the user for context (like regions), and add it into prompts.
5. Maybe questions regarding hosting this web app.
    a. For basic approach we can host it anywhere as there no API cost involved (that's why let's try to have a really good non AI algorithm for demojifying as that means almost no hosting cost)
6. How do we make sure that we have least production and deployment cost.
7. Development time

8. How to evaluate the results of translations?
    a. Human evaluation → have a rating bar after each translation for user
    b. AI Evaluation
    c.
9. Should we consider storing the history of the translation for each user?
    a. Depends on the use case, if the users are people who use this app to translate regular conversation then it's good to have a record but for now i believe let's not store them because even if we store we can only store one side of the chat (that means what the receiver received, which won't provide complete context to understand entire conversation, as the sender won't share his/her side)
    b. Agree. To keep it easy, no need to consider that now
10. How to store data > are we allowed to use database?
    a. What kind of data do we need to store?
11. Should we focus on the most common emojis used since there are many many emojis in 2025?
    a. I am not sure about java but python have these libraries who regularly update their emoji set (not sure how frequently but assuming it is more frequent than any other library)
12. How can we take accessibility into consideration?
    a. In this case, we have emojis which have various colors
    b. I can't think of anything else, but something to keep in mind
13. Basic social questions : like : how did you guys decide who will do what part of the project, on what bases you set your project timeline etc.
    a. Clusters: Front-end and back end
    b. Within Clusters we can maybe have mini clusters in which each mini cluster can try a different approach
        i. Since this is a warmup so the scale is not complex, but when coming to real project, maybe we can group by features
    c. Since we will be presenting on Thursday we can meet again perhaps **Wednesday 3pm**?
14. How do we set the structure of the frontend and backend, so that everyone can contribute together?
    a. How to handle Concurrent development without overlapping work or doing redundant work?

FRONT END

- Audria
- Andrew

BACK END

- Development
    - Bobby
    - Yuki Wu

        - Sriraksha
        - Aditya
        - Hanis
    - Research/Algorithm
        - Abhishek Dhaka
        - Rui
        - Gabrielle
        - Tongfei

PRESENTATION (Including presenting, creating slides and have a broad idea of entire project)

- Audria
- <rest of the team>

CLOUD (deploying the project once developed)

- Abhishek Dhaka
- Bobby

MANAGEMENT (progress manager, git manager):

- Khang
- Abhishek Dhaka