# Daily Todo AI Component

* every new day when you hit the site, the code copies over the previous day’s todos today
  + at the same time we want the AI to generate a new message for you based on all your todo history, you mission/goals, and your explanation of why each todo is important to your mission and goals
* hit homepage, get 7 days of todos
  + if it’s a new day, copy over previous day to new days, with blank checkboxes so you can start again
  + if it’s a new day - send flag to react to request a new AI message for the new day
    - react hits python with
      * user id
      * day number / date
      * new\_day flag true/false
* new python
  + will hit golang app to get any needed context about the user from the db
    - mission/goals
    - todo history
    - how each todo is important to mission/goals
    - daily notes/comments on each todo
  + then hit openai assistants, and continue or start convo thread
    - and just ask what is the encouraging message for the morning of today june 23 2024
    - stream response to react
    - also save to db, because on page refresh, react will not see “new\_day” true, but will still want to show the most recent ai encouragement message
* new database tables
  + user\_threads: mapping of user id to convo thread id
    - id, user\_id, thread\_id, created\_at
    - any given user can have multiple thread ids
  + user\_missions: the user’s mission in life
    - id, user\_id, text, created\_at, updated\_at
  + user\_todo\_relevance\_descriptions
    - general relevance description why each todo is important to the missions
    - id, todo or daily\_todo\_id, description,created\_at, updated\_at
  + user\_todo\_notes
    - notes about each todo that day, and how it went
    - add column to DailyTodos. default ‘’
  + saved\_assistant\_messages
    - save the assistant message so we can show it on the UI every request
    - id, user\_id, day, message, created\_at
* new golang
  + python will hit golang to get data about the user. could start with just a nicely formatted json file with everything about the user. see if that is enough for AI to generate good responses
* LATER
  + what if user wants to have back and forth chat with the assistant?
    - we can let user type, it just adds to thread and we get assistant response and display it
      * we put our own prompt of course 1-2x a day to get encouragement in the same thread
      * tricky part is saving all the questions and responses, and re-displaying them
      * v2 feature?
* next steps
  + rename all database tables to lowercase users, daily\_todos
  + create new database tables
    - user\_threads
      * empty
    - user\_missions
      * fill it in manually
      * add UI to enter/edit mission on user profile page in the future
    - saved\_assistant\_messages
      * id, user\_id, day, message, created\_at
  + new golang endpoints
    - fetch all user daily todo history, fetch mission, return nicely formatted and labeled json
    - fetch latest saved message from db for the user and return to python
      * react could also hit this, makes more sense that way actually
        + react hits golang for db, react hits python for new ai
        + python hits golang for db too
  + python
    - openai, instructor
    - have a create assistant python script, run once
    - webapp python script
      * if it’s new day
        + web app fetches that assistant (previously created)
        + gets latest user data from golang endpoint
        + sends user data and prompt (“what is today’s encouraging message”) to assistant. here is the user history as json
        + get response, and stream it to react
        + save response to db (how to do this while streaming?)
  + react
    - if new day hit python for new ai message
    - if not new day, hit golang for saved message