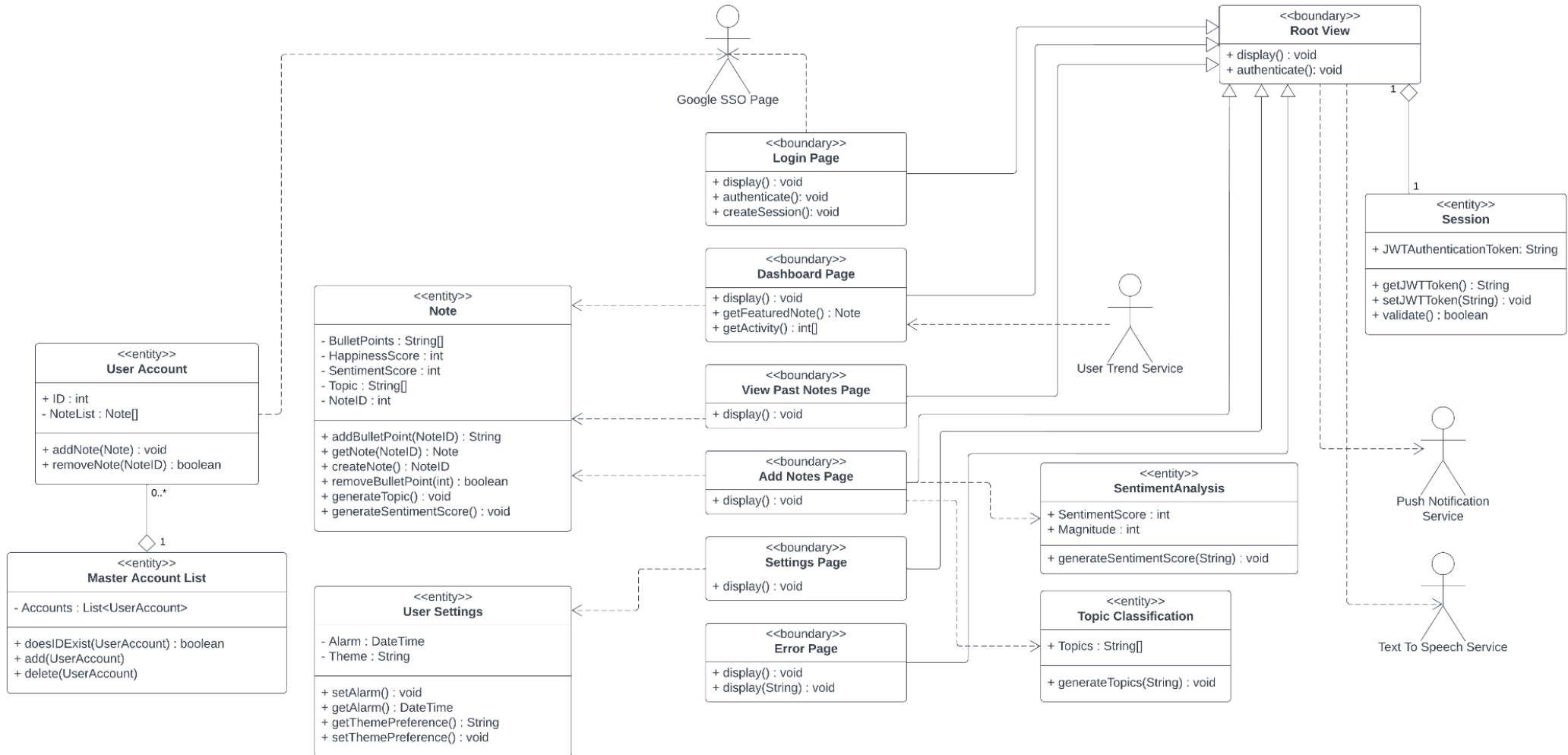## Selection of framework/architecture

- Webpage Hosting
  - o [Google Cloud Run (Website Example)](#)
    - § We will use docker to containerize our front-end tech stack for the desktop version, and host this by deploying it onto cloud run. By doing so, we get the scalability, availability, and security for which our application needs.

- Front-end
  - o [React & Ionic Framework](#)
    - § Our application requires multi-platform support, and as such, Ionic provides us the tools to cross-compile and utilize one codebase while providing a native-feel across all of the different platforms. As Ionic can integrate with multiple front-end frameworks, we've decided to choose React as our application will be highly component-based, which React tends to perform better with.

- Back-end
  - o [Google Cloud Run (API Example)](#)
    - § We will use Docker to containerize our API, and then build & deploy it via Google Cloud Run, as it allows us to be hassle-free of scalability concerns and it is fully managed.

- Machine-Learning Services
  - o Sentiment Analysis
    - § [Natural Language API (Analyzing Sentiment)](#)
      - · Google Cloud allows us to send a string (or from Cloud Storage) to their Natural Language AI and produces a sentiment score and magnitude.
      - ·
  - o Speech-To-Text
    - § [Cloud Speech-To-Text](#)
      - · To enable Speech-To-Text on computers, and devices without a built-in live transcription, we will use Google Cloud's Speech-To-Text API to send in voice and retrieve text from the end user.
  - o Topic Modeling / Text Classification
    - § [Natural Language API (Analyzing Entities)](#)
      - · Google Cloud also allows us to analyze topics within a string which can be stored and utilized for our Entity Encouragement Service to encourage users to input different entries.
    - § [Natural Language API (Analyzing Entity Sentiment)](#)
      - · We can use this, in order to draw up relations for our trend service within our application and draw potential connections between an entity and their sentiment value for users.

- Authentication Services
    - Identity Management (Firebase Authentication)
        - § As our application desires to have Google SSO, we've decided to use Firebase Authentication. There exists an integration within the SDK to work with React (our front-end).

- Database Services
    - Images / Large Data Objects (Voice/Video)
        - § Cloud Storage (Google)
            - We plan to use storage buckets as our primary choice of storing user voice/video/image data, as they are incredibly scalable, and the infrastructure is fully managed by google. Furthermore, it has fine-grained permissions to ensure that the data is held securely. It also can easily integrate with our REST API, as itself is also accessed via REST API. Moreover, it has an integration with Firebase Authentication
    - User Data (Notes)
        - § Firestore (Google Firebase)
            - A Serverless No-SQL Database, as our user data will tend to be unstructured data, that is more suited towards a document-based data type.

- Version Control
    - GitHub
        - We are deciding to use GitHub for source control, as it is quite easy to learn and has plenty of documentation and support for any issue that we encounter. Furthermore, there are numerous plugins that allow for seamless integration between our repository and deployment. In addition, it allows us to neatly organize our applications in separate repositories and group it under an organization for free.

Static Class Diagram (including attributes and operations, OO version)

Google SSO Page

<<boundary>>
Root View
+ display() : void
+ authenticate(): void

<<boundary>>
Login Page
+ display() : void
+ authenticate(): void
+ createSession(): void

<<entity>>
Session
+ JWTAuthenticationToken: String
+ getJWTToken() : String
+ setJWTToken(String) : void
+ validate() : boolean

<<boundary>>
Dashboard Page
+ display() : void
+ getFeaturedNote() : Note
+ getActivity() : int[]

User Trend Service

<<entity>>
Note
- BulletPoints : String[]
- HappinessScore : int
- SentimentScore : int
- Topic : String[]
- NoteID : int
+ addBulletPoint(NoteID) : String
+ getNote(NoteID) : Note
+ createNote() : NoteID
+ removeBulletPoint(int) : boolean
+ generateTopic() : void
+ generateSentimentScore() : void

<<entity>>
User Account
+ ID : int
- NoteList : Note[]
+ addNote(Note) : void
+ removeNote(NoteID) : boolean

<<boundary>>
View Past Notes Page
+ display() : void

<<boundary>>
Add Notes Page
+ display() : void

<<entity>>
SentimentAnalysis
+ SentimentScore : int
+ Magnitude : int
+ generateSentimentScore(String) : void

Push Notification Service

<<entity>>
Master Account List
- Accounts : List<UserAccount>
+ doesIDExist(UserAccount) : boolean
+ add(UserAccount)
+ delete(UserAccount)

<<boundary>>
Settings Page
+ display() : void

<<entity>>
Topic Classification
+ Topics : String[]
+ generateTopics(String) : void

<<entity>>
User Settings
- Alarm : DateTime
- Theme : String
+ setAlarm() : void
+ getAlarm() : DateTime
+ getThemePreference() : String
+ setThemePreference() : void

<<boundary>>
Error Page
+ display() : void
+ display(String) : void

Text To Speech Service

0..*   1   1   1

Login

BASIC COURSE:

On the Login Page, the User clicks Sign-in with Google Button, and the system displays Google SSO Page. The User types Password for their Google Account. Then, the User clicks Google SSO Next Button, where Google Identity generates authentication cookies on a valid login. After cookie generation, the system receives the cookies from Google Identity and displays Gratitude Notes Dashboard Page.
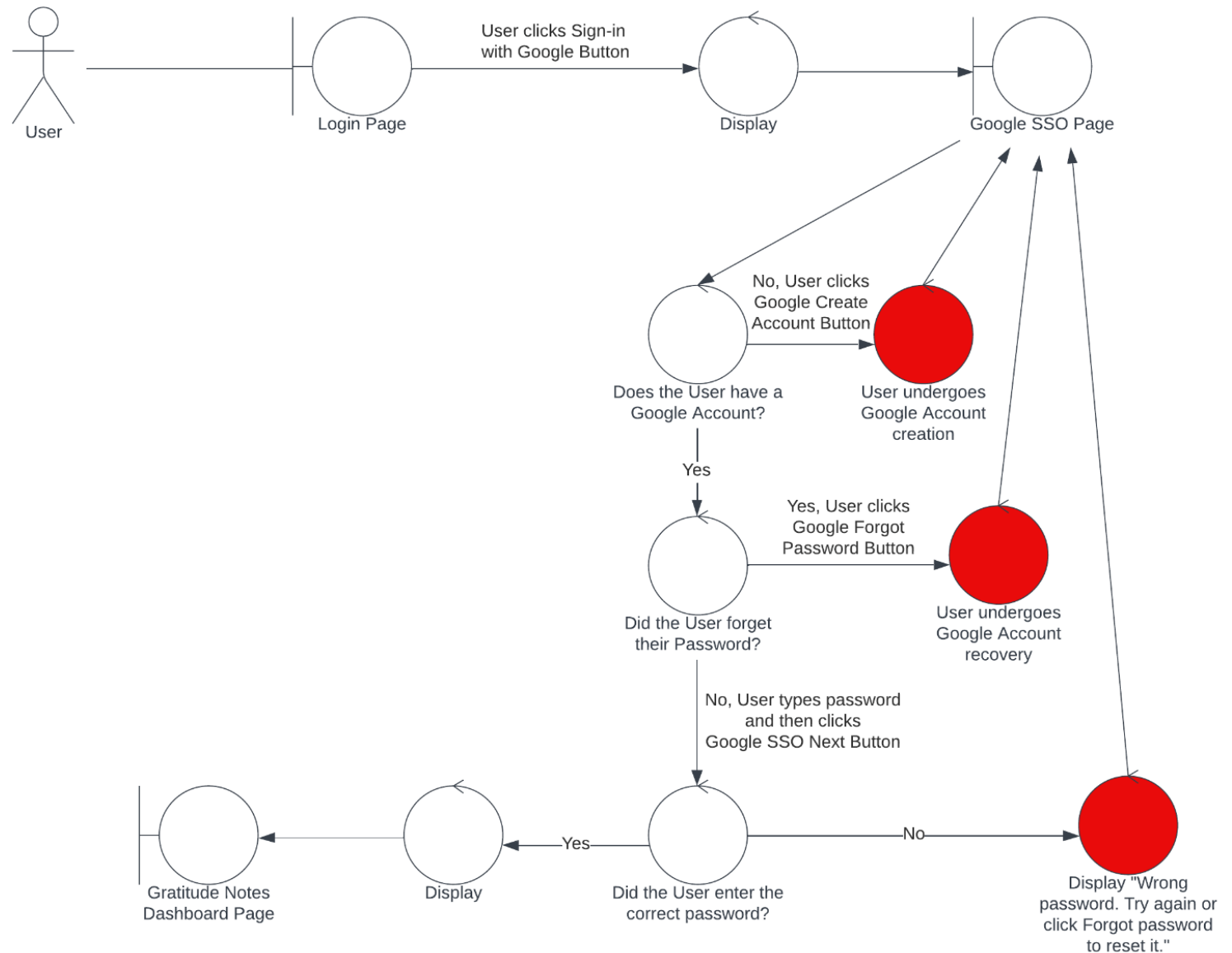
ALTERNATE COURSES:

User does not have a Google Account or they want to create a Google Account: Then, the User clicks Google Create Account Button. Next, the User undergoes Google Account creation. After verification, the system displays Gratitude Notes Dashboard Page.

User forgot their Password: Then, the User clicks Google Forgot Password Button. Next, the User undergoes Google Account recovery. After verification, the system displays Gratitude Notes Dashboard Page.

User enters wrong Password: Then, Google displays Google Wrong Password message, which contains "Wrong password. Try again or click Forgot password to reset it."

User

Login Page

User clicks Sign-in with Google Button

Display

Google SSO Page

Does the User have a Google Account?

No, User clicks Google Create Account Button

User undergoes Google Account creation

Yes

Did the User forget their Password?

Yes, User clicks Google Forgot Password Button

User undergoes Google Account recovery

No, User types password and then clicks Google SSO Next Button

Did the User enter the correct password?

Yes

No

Display

Gratitude Notes Dashboard Page

Display "Wrong password. Try again or click Forgot password to reset it."

Login

Author: Brady Waughen

BASIC COURSE:

On the Login Page, the User clicks Sign-in with Google Button, and the system displays Google SSO Page. The User types Password for their Google Account. Then, the User clicks Google SSO Next Button, where Google Identity generates authentication cookies on a valid login. After cookie generation, the system receives the cookies from Google Identity and displays Gratitude Notes Dashboard Page.
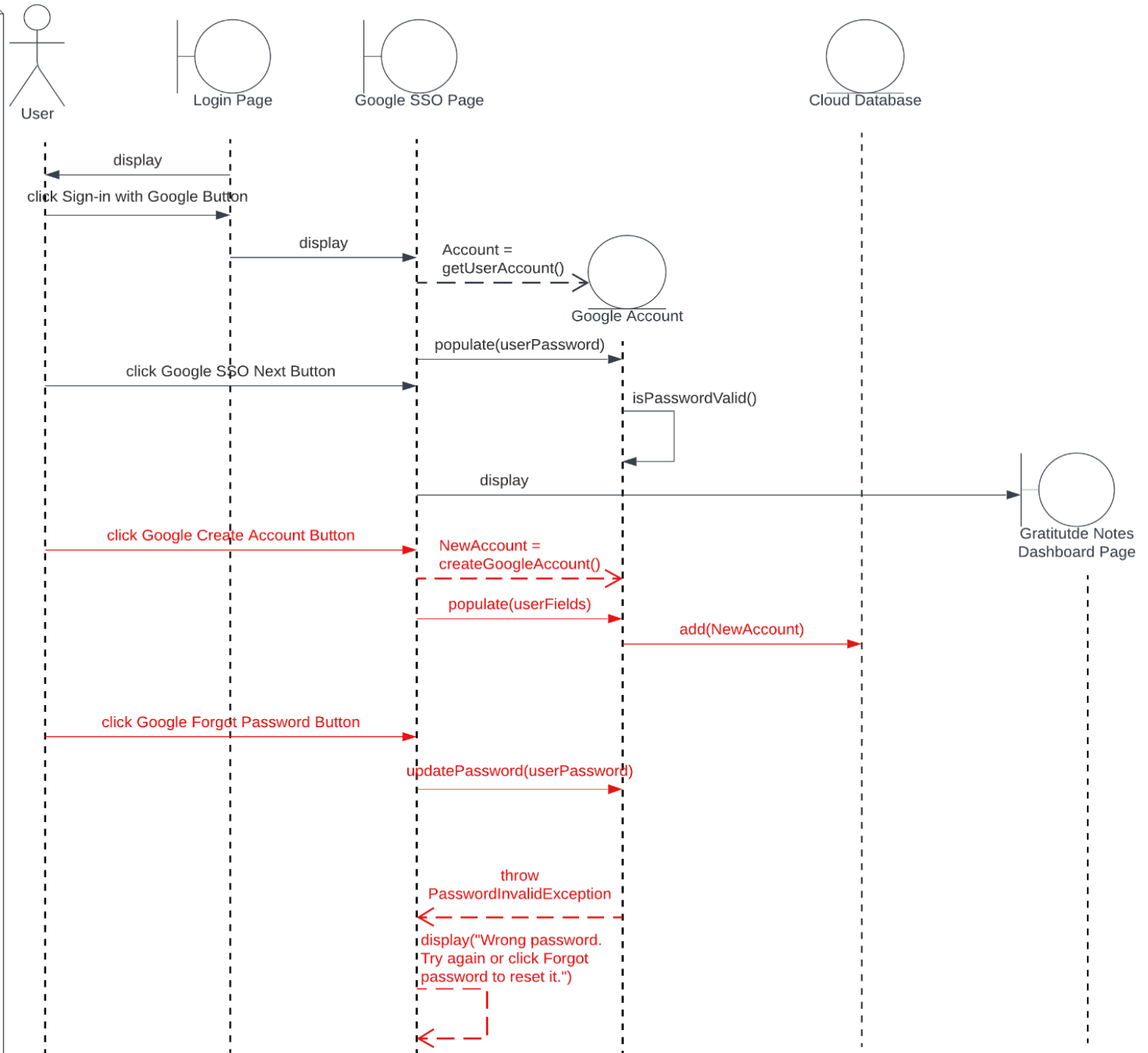
ALTERNATE COURSES:

User does not have a Google Account or they want to create a Google Account: Then, the User clicks Google Create Account Button. Next, the User undergoes Google Account creation. After verification, the system displays Gratitude Notes Dashboard Page.

User forgot their Password: Then, the User clicks Google Forgot Password Button. Next, the User undergoes Google Account recovery. After verification, the system displays Gratitude Notes Dashboard Page.

User enters wrong Password: Then, Google displays Google Wrong Password message, which contains "Wrong password. Try again or click Forgot password to reset it."

Participants: User · Login Page · Google SSO Page · Cloud Database · Google Account · Gratitude Notes Dashboard Page

Messages:

- display
- click Sign-in with Google Button
- display
- Account = getUserAccount()
- populate(userPassword)
- click Google SSO Next Button
- isPasswordValid()
- display
- click Google Create Account Button
- NewAccount = createGoogleAccount()
- populate(userFields)
- add(NewAccount)
- click Google Forgot Password Button
- updatePassword(userPassword)
- throw PasswordInvalidException
- display("Wrong password. Try again or click Forgot password to reset it.")

Author: Allison Moyer

Add Notes

BASIC COURSE:

On the Gratitude Notes Dashboard Page, the User clicks the Add Button. The system displays Add Notes Page. The User enters their Note Entry and their Gratitude Happiness Score. After, the User clicks the Submit Button; the system checks the Note Entry does not exceed 280 words and the Happiness Score. Then, the system sends the Note Entry to the Entry Encouragement Service. After the check, the system updates the Cloud Database and the system sends the Note Entry to the Text Classification Service and Sentiment Analysis Service. Lastly, the system displays the Gratitude Notes Dashboard Page.
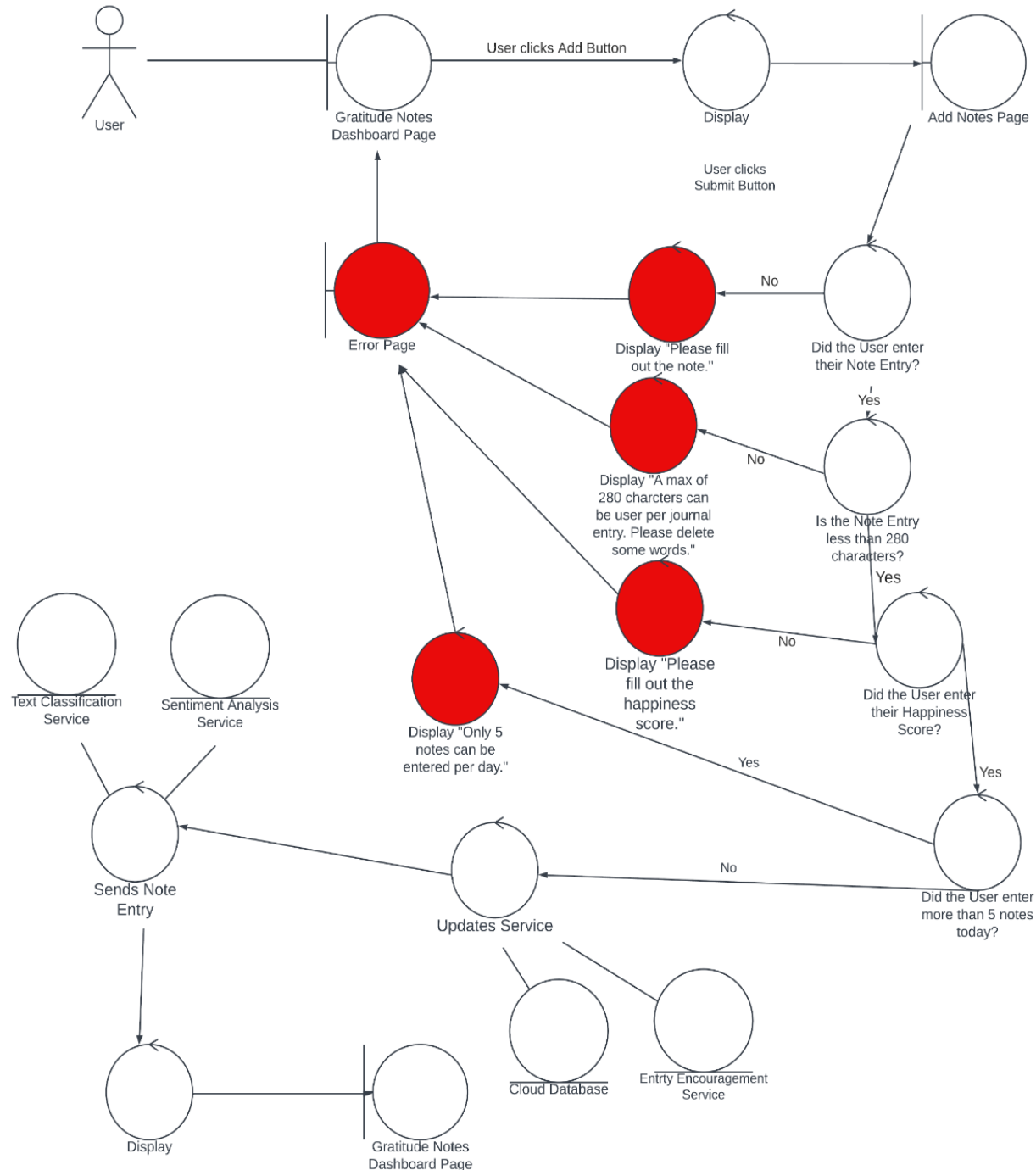
ALTERNATE COURSES:

User hits the submit button without entering a note: The system shows an Error Page, which contains, "Please fill out the note." Then, the system displays Gratitude Notes Dashboard Page.

User hits the submit button without entering a happiness score: The system shows an Error Page, which contains, "Please fill out the happiness score." Then, the system displays Gratitude Notes Dashboard Page.

User fills out more than 5 notes: The system shows an Error Page which contains a message, "Only 5 notes can be entered per day." Then, the system displays Gratitude Notes Dashboard Page.

The Note Entry exceeds more than 280 characters: The system shows an Error Page, which contains a message, "A max of 280 characters can be used per journal entry. Please delete some words." Then, the system displays the Gratitude Notes Dashboard Page.

Add Notes

BASIC COURSE:

On the Gratitude Notes Dashboard Page, the User clicks the Add Button. The system displays Add Notes Page. The User enters their Note Entry and their Gratitude Happiness Score. After, the User clicks the Submit Button; the system checks the Note Entry does not exceed 280 words and the Happiness Score. Then, the system sends the Note Entry to the Entry Encouragement Service. After the check, the system updates the Cloud Database and the system sends the Note Entry to the Text Classification Service and Sentiment Analysis Service. Lastly, the system displays the Gratitude Notes Dashboard Page.
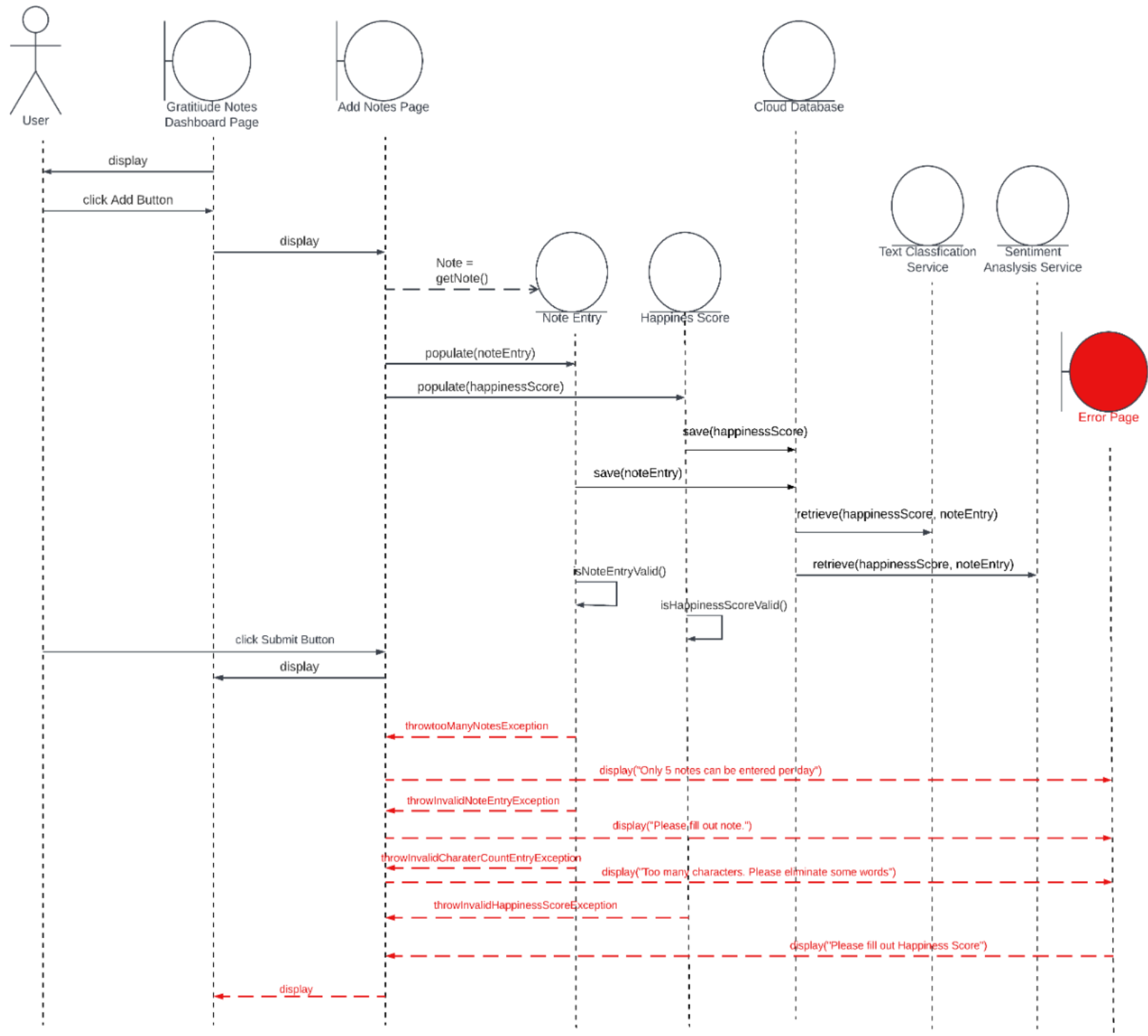
ALTERNATE COURSES:

User hits the submit button without entering a note: The system shows an Error Page, which contains, "Please fill out the note." Then, the system displays Gratitude Notes Dashboard Page.

User hits the submit button without entering a happiness score: The system shows an Error Page, which contains, "Please fill out the happiness score." Then, the system displays Gratitude Notes Dashboard Page.

User fills out more than 5 notes: The system shows an Error Page which contains a message, "Only 5 notes can be entered per day." Then, the system displays Gratitude Notes Dashboard Page.

The Note Entry exceeds more than 280 characters: The system shows an Error Page, which contains a message, "A max of 280 characters can be used per journal entry. Please delete some words." Then, the system displays the Gratitude Notes Dashboard Page.

Participants (lifelines):
- User
- Gratitude Notes Dashboard Page
- Add Notes Page
- Note Entry
- Happiness Score
- Cloud Database
- Text Classification Service
- Sentiment Analysis Service
- Error Page

Messages:
- display (Gratitude Notes Dashboard Page → User)
- click Add Button (User → Gratitude Notes Dashboard Page)
- display (Gratitude Notes Dashboard Page → Add Notes Page)
- Note = getNote() (Add Notes Page → Note Entry)
- populate(noteEntry) (Add Notes Page → Note Entry)
- populate(happinessScore) (Add Notes Page → Happiness Score)
- save(happinessScore) (Happiness Score → Cloud Database)
- save(noteEntry) (Note Entry → Cloud Database)
- retrieve(happinessScore, noteEntry) (Cloud Database → Text Classification Service)
- retrieve(happinessScore, noteEntry) (Cloud Database → Sentiment Analysis Service)
- isNoteEntryValid() (Note Entry self-call)
- isHappinessScoreValid() (Happiness Score self-call)
- click Submit Button (User → Gratitude Notes Dashboard Page)
- display (Gratitude Notes Dashboard Page → Add Notes Page)
- throwtooManyNotesException (Error Page → Add Notes Page)
- display("Only 5 notes can be entered per day")
- throwInvalidNoteEntryException
- display("Please fill out note.")
- throwInvalidCharaterCountEntryException
- display("Too many characters. Please eliminate some words")
- throwInvalidHappinessScoreException
- display("Please fill out Happiness Score")
- display (Add Notes Page → Gratitude Notes Dashboard Page)

View Past Notes Robustness Diagram

Author: Myan Nguyen
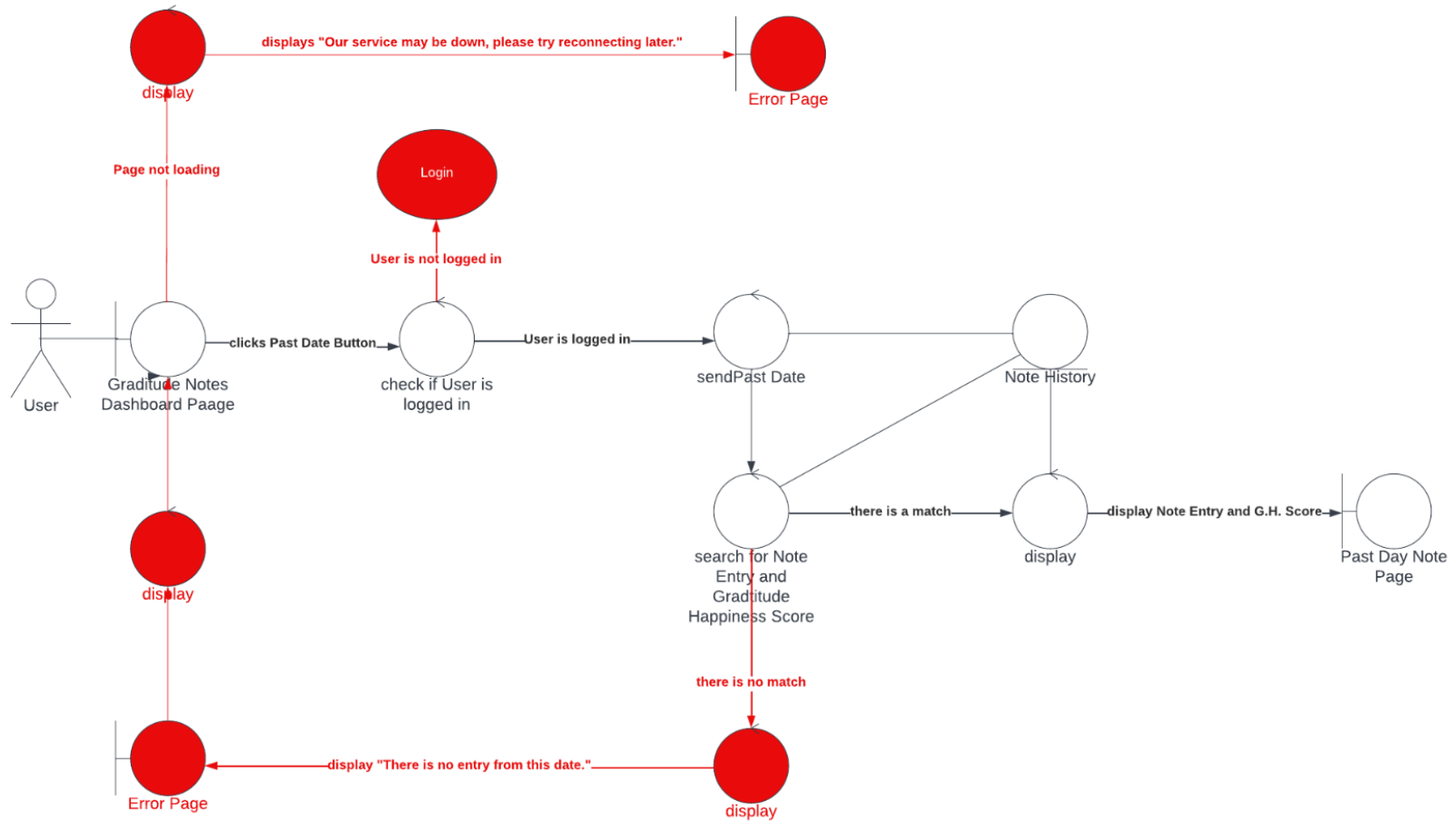
View Past Note Use Case Text
BASIC COURSE:

On the Gratitude Notes Dashboard Page, the User clicks Past Date Button from Calendar Widget. The system receives Past Date and the system searches for Note Entry and Gratitude Happiness Score from Note History. The system displays Note Entry and Gratitude Happiness Score on the Past Day Note Page.

ALTERNATE COURSES:

Page not loading: The system shows Error Page, which contains a message, "Our service may be down, please try reconnecting later."

A past date has no matching Gratitude Note/Happiness Score: The system shows Error Page, which contains a message, "There is no entry from this date." Then, the system displays Gratitude Notes Dashboard Page.
The User is not logged in: The session invokes the Login Use Case.

display

displays "Our service may be down, please try reconnecting later."

Error Page

Page not loading

Login

User is not logged in

User

clicks Past Date Button

Gratitude Notes Dashboard Paage

User is logged in

check if User is logged in

sendPast Date

Note History

display

search for Note Entry and Gradtitude Happiness Score

there is a match

display

display Note Entry and G.H. Score

Past Day Note Page

there is no match

display

Error Page

display "There is no entry from this date."

display

View Past Notes Sequence Diagram

Author: Myan Nguyen

View Past Note Use Case Text
BASIC COURSE:

On the Gratitude Notes Dashboard Page, the User clicks Past Date Button from Calendar Widget. The system receives Past Date and the system searches for Note Entry and Gratitude Happiness Score from Note History. The system displays Note Entry and Gratitude Happiness Score on the Past Day Note Page.

ALTERNATE COURSES:

Page not loading: The system shows Error Page, which contains a message, "Our service may be down, please try reconnecting later."

A past date has no matching Gratitude Note/Happiness Score: The system shows Error Page, which contains a message, "There is no entry from this date." Then, the system displays Gratitude Notes Dashboard Page.
The User is not logged in: The session invokes the Login Use Case.

User

Graditude Notes Dashboard Page

Session

Note History

display()

click Past Date Button

isUserLoggedIn()

searchForEntry(pastDate)

display(noteEntry, GHScore)

Past Day Note Page

Page not loading

display("Our service may be down, please try reconnecting later.")

Error Page

A past date has no matching Gratitude Note/ Happiness Score

display("There is no entry from this date.")

Error Page

display()

User is not logged in

User not logged in
The session invokes Login use case