CEN 4010 Principles of Software Engineering 2.1

Milestone 4: Beta Launch and Final Project Reviews

Gianluca: gguagliardo2017@fau.edu; Leon: lrice2019@fau.edu;
Mayte: mramirezcald2018@fau.edu; Hendrien: hpaul2017@fau.edu;
Charlton: cbenting2019@fau.edu;

Revision History Table:	02/16/2021	Waiting In for instructor feedback...
Revision History Table:	04/16/2021	Revise Data Definition & High-Level Requirements
Revision History Table:	04/11/2021	Waiting In for instructor feedback...

Team:

Group Name: Alpha Technologies

Scrum Master: Gianluca Guagliardo

Product Owner: Leon Rice III

Roles:

Charlton Benting (Dev Team/Backend)

Gianluca Guagliardo (Dev Team/Backend)

Hendrein Paul (Dev Team/Frontend)

Mayte Ramirez-Calderon (Dev Team/Frontend Lead)

Leon Rice III (Dev Team/Backend Lead)

https://github.com/cen4010-s2021-g04/M4.git

Product Summary 2.2

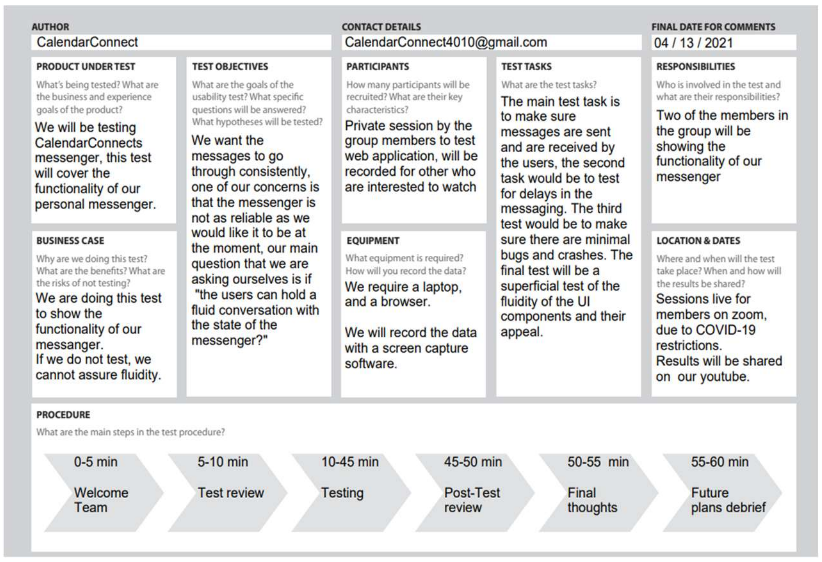
Product name: CalenderConnect

The main focus of Calendar Connect is to allow people to stay connected with their friends, family, coworkers, and loved ones. Our app integrates a person's schedule and allows them to share and connect with others to provide a clear visual of people's availability. Our user-friendly dashboard can be customized and organized to the user's needs and includes features such as to-do lists, groups, favorites, upcoming events, messages, alerts, and more! CalendarConnect allows users to create groups to share calendars with so you can maintain connection between family members, friend groups, school peers, or work colleagues. Our integration of pictures throughout your calendar month and year allows the app to become more personalized and for users to have fun and express themselves creatively. When you have an upcoming class, meeting, study session, or need some time alone, Calendar Connect can relay that to others to make sure you are not interrupted. On the other hand, when you are looking for free time within a group to spend some time together, whether virtually or in-person, Calendar Connect does the work of finding free time for the users so they can focus on planning and enjoying the time together. Unlike most apps that are tailored specifically for just work, school, or businesses, Calendar Connect offers a large reachability that can be used by everyone for all sorts of needs. Calendar Connect's integration allows users to customize their features to fit their needs either for school, work, personal life, or all of the above.

Functions:

- Create personal calendar
- Create group calendar
- Create new contact
- Create group chat message
- Send message to contact
- Ability to create personal and group calendar

Usability Test Plan 2.3



AUTHOR
CalendarConnect

CONTACT DETAILS
CalendarConnect4010@gmail.com

FINAL DATE FOR COMMENTS
04 / 13 / 2021

PRODUCT UNDER TEST
What's being tested? What are the business and experience goals of the product?
We will be testing CalendarConnects messenger, this test will cover the functionality of our personal messenger.

BUSINESS CASE
Why are we doing this test? What are the benefits? What are the risks of not testing?
We are doing this test to show the functionality of our messanger.
If we do not test, we cannot assure fluidity.

TEST OBJECTIVES
What are the goals of the usability test? What specific questions will be answered? What hypotheses will be tested?
We want the messages to go through consistently, one of our concerns is that the messenger is not as reliable as we would like it to be at the moment, our main question that we are asking ourselves is if "the users can hold a fluid conversation with the state of the messenger?"

PARTICIPANTS
How many participants will be recruited? What are their key characteristics?
Private session by the group members to test web application, will be recorded for other who are interested to watch

EQUIPMENT
What equipment is required? How will you record the data?
We require a laptop, and a browser.

We will record the data with a screen capture software.

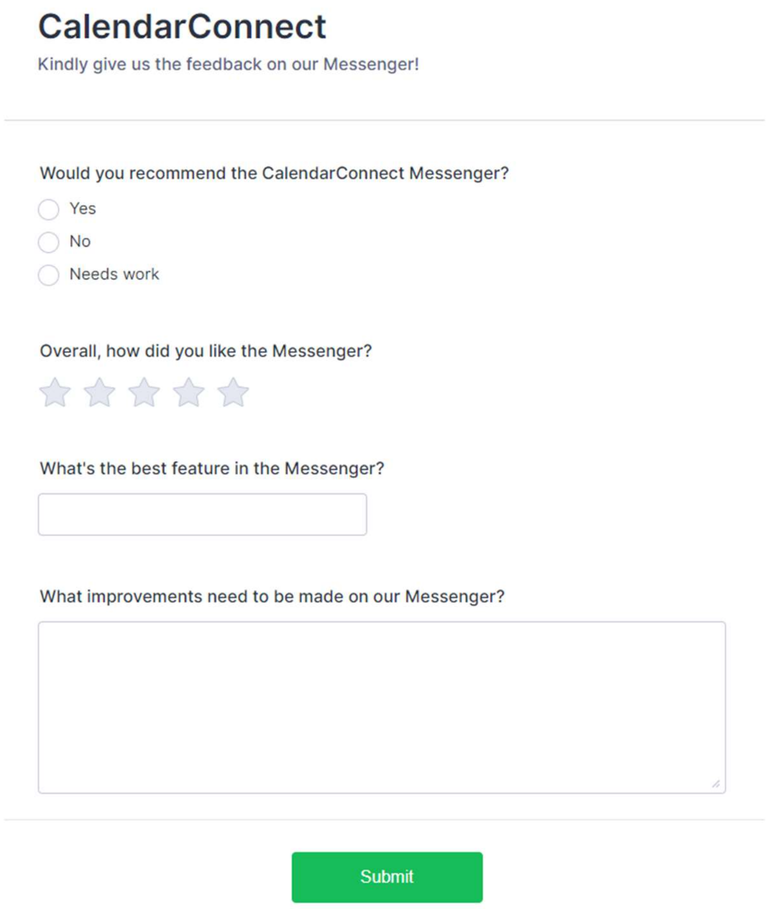
TEST TASKS
What are the test tasks?
The main test task is to make sure messages are sent and are received by the users, the second task would be to test for delays in the messaging. The third test would be to make sure there are minimal bugs and crashes. The final test will be a superficial test of the fluidity of the UI components and their appeal.

RESPONSIBILITIES
Who is involved in the test and what are their responsibilities?
Two of the members in the group will be showing the functionality of our messenger

LOCATION & DATES
Where and when will the test take place? When and how will the results be shared?
Sessions live for members on zoom, due to COVID-19 restrictions.
Results will be shared on our youtube.

PROCEDURE
What are the main steps in the test procedure?

0-5 min	5-10 min	10-45 min	45-50 min	50-55 min	55-60 min
Welcome Team	Test review	Testing	Post-Test review	Final thoughts	Future plans debrief



CalendarConnect

Kindly give us the feedback on our Messenger!

Would you recommend the CalendarConnect Messenger?

- ○ Yes
- ○ No
- ○ Needs work

Overall, how did you like the Messenger?

☆ ☆ ☆ ☆ ☆

What's the best feature in the Messenger?

What improvements need to be made on our Messenger?

https://form.jotform.com/211027832120137

QA Test Plan 2.4

- Test Objectives
  - Making sure messages send and receive.
  - Making sure group messages send and receive
  - Making sure all browsers work with code
  - Making sure adding multiple users to messages works.

- Hardware and software setup
  - Laptop, PC, smartphone
  - Browsers: Chrome, Edge, Firefox

- Feature to be tested
  - Send and Receive
  - Adding user with ID to habilitate private message
  - Receive group message
  - Create contact

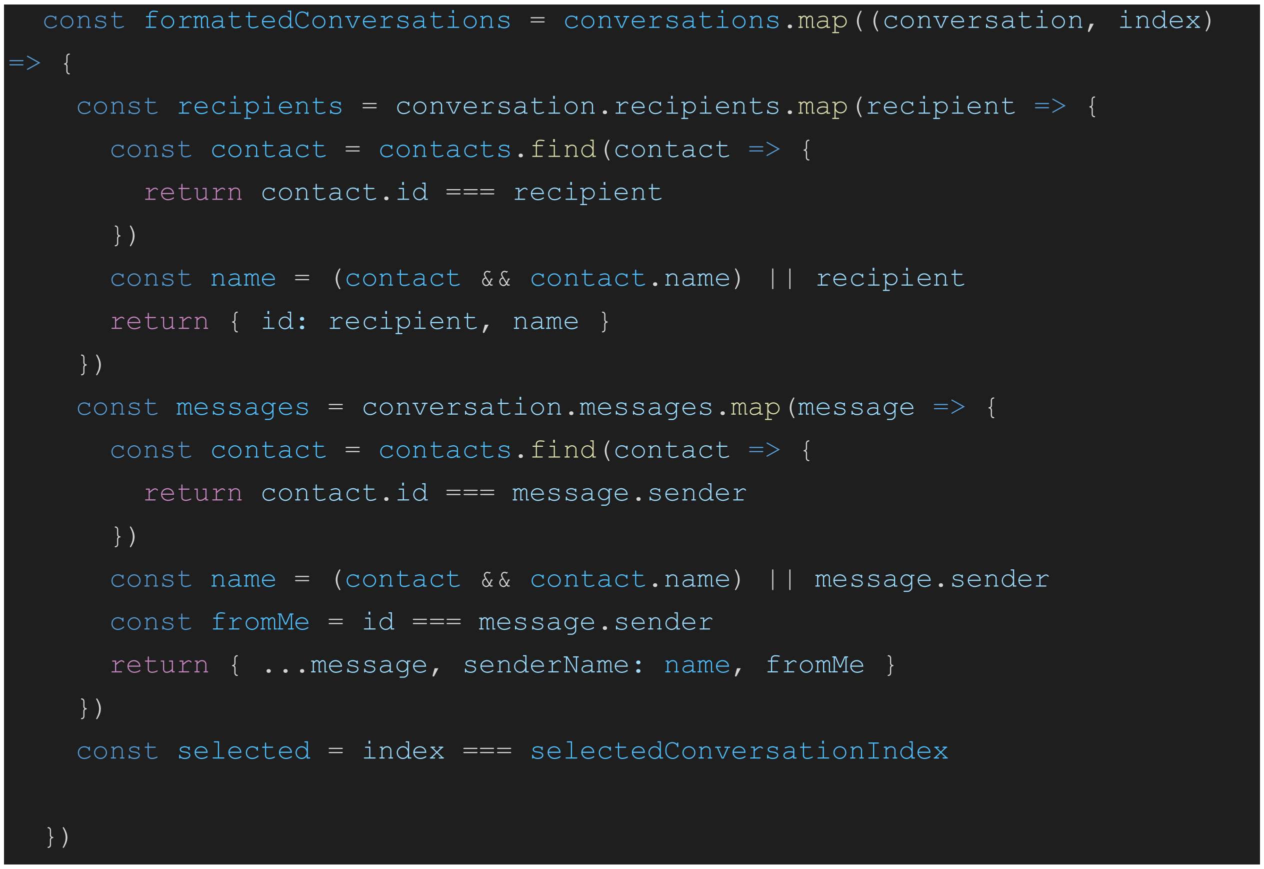
- Actual test cases
  - Adding Contact
  - Send-Receive
  - Sending Receiving group messages
  - Creating group messages

Test #	Test Title	Test Descriptions	Test input	Expected correct output	Test Results
1	Chrome - Send	Sending message	Test message 1	PASS	PASS
2	Chrome - Receive	Receiving Message	Test message 1	PASS	FAIL
3	Edge - Send	Sending message	Test message 2	PASS	PASS
4	Edge - Receive	Receiving Message	Test message 2	PASS	FAIL

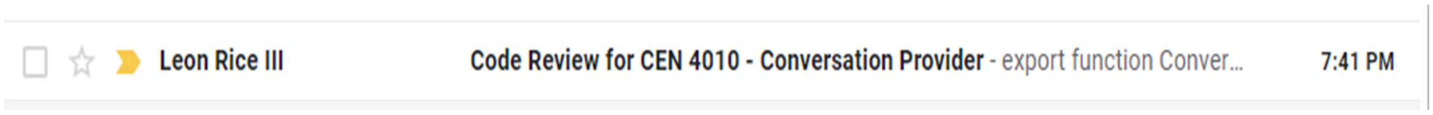
5	Firefox - Send	Sending message	Test message 3	PASS	PASS
6	Firefox - Receive	Receiving Message	Test message	PASS	FAIL
7	Create Group Message	Create a Group Message Board	Admin1, Admin2, Admin3	PASS	PASS
7	Send Group Message	Send Message to Multiple Contacts	Test Group Message 1	PASS	PASS
8	Receive Group Message	Receive Group Message	Test Group Message	PASS	FAIL
9	Create Contact	Creating a new contact	New contact User ID	PASS	PASS

Code Review 2.5

```
export function ConversationsProvider({ id, children }) {
  const [conversations, setConversations] =
useLocalStorage('conversations', [])
  const [selectedConversationIndex, setSelectedConversationIndex] =
useState(0)
  const { contacts } = useContacts()
  const socket = useSocket()

  function createConversation(recipients) {
    setConversations(prevConversations => {
      return [...prevConversations, { recipients, messages: [] }]
    })
  }
  const addMessageToConversation = useCallback(({ recipients, text, sender
}) => {
    setConversations(prevConversations => {
      let madeChange = false
      const newMessage = { sender, text }
      const newConversations = prevConversations.map(conversation => {
        if (arrayEquality(conversation.recipients, recipients)) {
          madeChange = true
          return {
            ...conversation,
            messages: [...conversation.messages, newMessage]
          }
        }
        return conversation
      })
      if (madeChange) {
        return newConversations
      } else {
        return [
          ...prevConversations,
          { recipients, messages: [newMessage] }
        ]
      }
    })
  }, [setConversations])
  useEffect(() => {
    if (socket == null) return

    socket.on('receive-message', addMessageToConversation)

    return () => socket.off('receive-message')
  }, [socket, addMessageToConversation])

  function sendMessage(recipients, text) {
    socket.emit('send-message', { recipients, text })

    addMessageToConversation({ recipients, text, sender: id })
  }
```



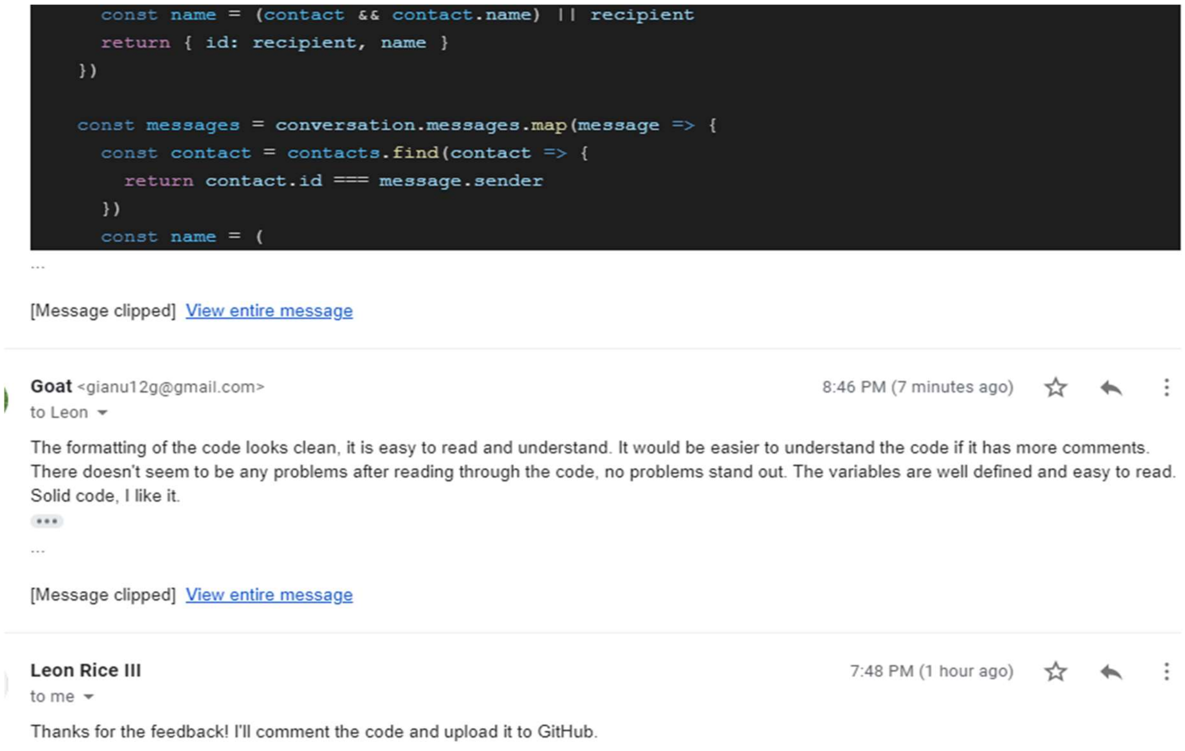
```
  const formattedConversations = conversations.map((conversation, index)
=> {
    const recipients = conversation.recipients.map(recipient => {
      const contact = contacts.find(contact => {
        return contact.id === recipient
      })
      const name = (contact && contact.name) || recipient
      return { id: recipient, name }
    })
    const messages = conversation.messages.map(message => {
      const contact = contacts.find(contact => {
        return contact.id === message.sender
      })
      const name = (contact && contact.name) || message.sender
      const fromMe = id === message.sender
      return { ...message, senderName: name, fromMe }
    })
    const selected = index === selectedConversationIndex

  })
```



Leon Rice III — **Code Review for CEN 4010 - Conversation Provider** - export function Conver... — 7:41 PM

Code was submitted by group member Leon Rice III

Peer Reviewer: Gianluca Guagliardo



```
      const name = (contact && contact.name) || recipient
      return { id: recipient, name }
    })

    const messages = conversation.messages.map(message => {
      const contact = contacts.find(contact => {
        return contact.id === message.sender
      })
      const name = (
```

...

[Message clipped] View entire message

Goat <gianu12g@gmail.com> — 8:46 PM (7 minutes ago)
to Leon

The formatting of the code looks clean, it is easy to read and understand. It would be easier to understand the code if it has more comments. There doesn't seem to be any problems after reading through the code, no problems stand out. The variables are well defined and easy to read. Solid code, I like it.

...

[Message clipped] View entire message

Leon Rice III — 7:48 PM (1 hour ago)
to me

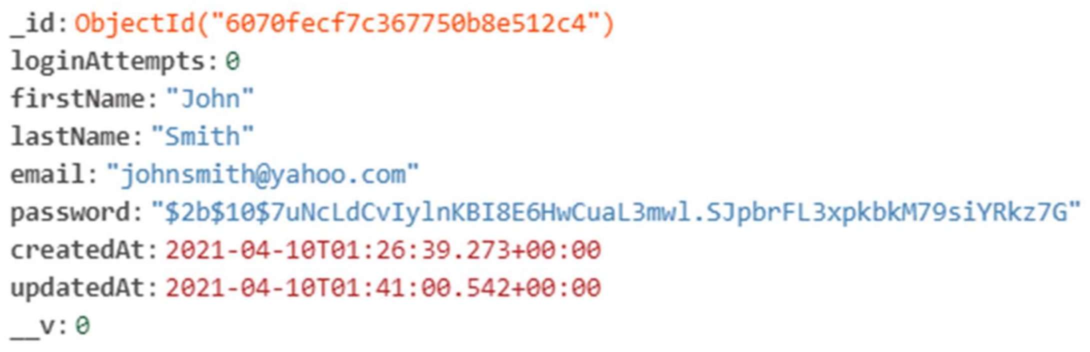
Thanks for the feedback! I'll comment the code and upload it to GitHub.

The piece of code reviewed is pretty essential in the functionality of the test plan function, so it was important to have the team review and comment on it. It will be posted on GITHUB with comments explaining basic functionality.

https://github.com/cen4010-s2021-g04/M4.git

Self-check on best security practices 2.6

The major assets we're protecting are the users' name, email, password, phone number, date of birth, and calendar.



```
_id: ObjectId("6070fecf7c367750b8e512c4")
loginAttempts: 0
firstName: "John"
lastName: "Smith"
email: "johnsmith@yahoo.com"
password: "$2b$10$7uNcLdCvIylnKBI8E6HwCuaL3mwl.SJpbrFL3xpkbkM79siYRkz7G"
createdAt: 2021-04-10T01:26:39.273+00:00
updatedAt: 2021-04-10T01:41:00.542+00:00
__v: 0
```

When registering the first and last name, email, and password is being validated via the code below. We used the node module validator to effectively validate the user form input. (register.js)

```
function validateRegistration(data) {

    let errors = {};


    data.firstName = !isEmpty(data.firstName) ? data.firstName : '';
    data.lastName = !isEmpty(data.lastName) ? data.lastName : '';
    data.email = !isEmpty(data.email) ? data.email : '';
    data.password = !isEmpty(data.password) ? data.password : '';
    data.posswordConfim = !isEmpty(data.passwordConfirm) ?
data.passwordConfirm : '';


    if (validator.isEmpty(data.firstName))
    {
        errors.firstName = 'First name required';
    }

    if (validator.isEmpty(data.lastName))
    {
        errors.lastName = 'Last name required';
    }


    if (validator.isEmpty(data.email))
    {
        errors.email = 'Email required';
    }
    else if (!validator.isEmail(data.email))
    {
        errors.email = 'Invalid email';
    }


    if (validator.isEmpty(data.password))
    {
        errors.password = 'Password required';
    }


    if (validator.isEmpty(data.passwordConfirm))
    {
        errors.passwordConfirm = 'Confirmation password required';
    }


    if (!validator.isLength(data.password, { min: 8, max: 32 }))
    {
        errors.password = 'Password length must be 8-32 charcters long';
    }



    if (!validator.equals(data.password, data.passwordConfirm))
    {
        errors.passwordConfirm = 'Passwords must match';
    }

    return { errors, valid: isEmpty(errors) };
};
```

When logging in email and password is being validated via the code below. We used the node module validator to effectively validate the user form input. (login.js)

```
function validateLogin(data) {
    let errors = {};

    data.email = !isEmpty(data.email) ? data.email : '';
    data.password = !isEmpty(data.password) ? data.password : '';

    if (validator.isEmpty(data.email))
    {
        errors.email = 'Email required';
    }
    else if (!validator.isEmail(data.email))
    {
        errors.email = 'Invalid email';
    }



    if (validator.isEmpty(data.password))
    {
        errors.password = 'Password required';
    }
    return { errors, valid: isEmpty(errors) };
};
```

Self-check: Adherence to original Non-functional specs() 2.7

No.	Non-Functional Requirement Description	Status
FR0 1	**Usability** • Must have color blind mode, and dark mode for customizability and user necessity. (colors customizable) • Buttons must be large enough for users of all ages. (customizable) • Calendars will also be customizable for user comfort. (size, color, location) • Text will be customizable.	**On Track**
FR0 2	**Reliability / Availability**: • Stable website, no bugs, or crashes • Website cannot crash from too many users online at once(sometimes it's unavoidable) • In case of outage there will be back-up generators to give time to inform the users of an outage and estimated server downtime. • Servers will be owned and accessible by CalendarConnect to ensure upkeep and uptime.	**On Track**
FR0 3	**Scalability**: • The system will be upgraded according to the number of users • Server will be able to hold a greater number of users than the expected amount for lower latency, faster response times, and server overcrowding. • Physical server space will be sufficient for future upgrades.	**On Track**
FR0 4	**Performance**	**On Track**

	• Quick response time, low MS latency on clicks. • Smooth transitions(low MS and reactive movements) • When pinging a user, the ping should be near instant.(notification, calendar requests, etc.) • Functional buttons, users should not have to click more than once to access anything on CalendarConnect. (hitboxes)	
FR05	**Supportability** • Will have support for android and IOS (down the line) • CalendarConnect will have an email that will allow customers to send their perspectives to the company. This will allow for the Devs to make changes accordingly. • CalendarConnect will have a short informational video explaining how to use the website/app and its features.	**On Track**
FR06	**Security**: • Highly maintained security measures, especially since CalendarConnect deals with personal information such as locations and schedules. • Cloud and physical user information must also be protected at all times. Security measures involving these will move quickly and secretly.	**On Track**