





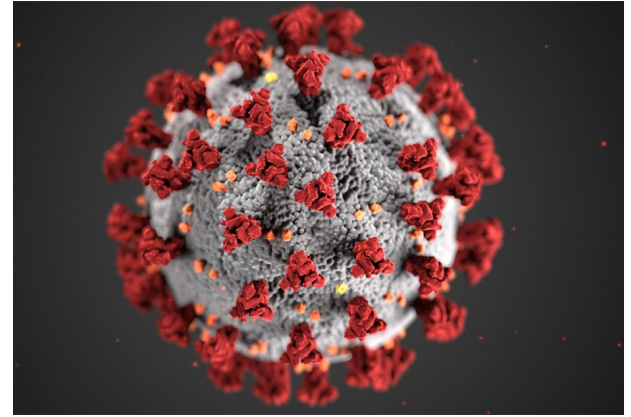
# Virus Validate

By Hanson Liu  
Ben Gagnon  
Feroze Attai  
Aramita Dhar



# Purpose of Software

- Allows for safe scheduling during pandemic
- Create/manage meetings and manage/invite guests if:
  - User authenticates as admin/employee
- Provides invited guest access to building if:
  - User authenticates as a guest AND
  - It is near the scheduled meeting time AND
  - The guest clears a COVID and Monkeypox health check



# Guest Storyboard

John is a client who is attending a meeting with a requirements engineer at the engineer's company headquarters. However, in order for John to attend the meeting, he must know the details of the meeting as well as receive authorization to enter the building.

John receives an invite email from the admin with a code that they can use to log into the app. This code will identify John inside the app. John downloads the app and inputs the guest code to log in. If the code is incorrect or John did not receive a code, the app will not allow him to log in.

Once John logs in, the app will redirect him to a survey. The system web scrapes the CDC website for covid and monkey pox symptoms, removing duplicate symptoms, and creates a form that evaluates if the guest has these symptoms.

John fills out and submits the form within a certain time frame relative to his appointment time (30 mins - 1 hr). If John's answers indicate an illness, the system flags their account so John can't attend the meeting and notifies both John and the admin. If the user's answers do not indicate an illness, their account is permitted for them to attend the meeting and they can unlock the door one time.

# Admin Storyboard

Jenn is an employee who needs to schedule a meeting with an external guest and monitor their status. She logs into the app and selects the '+' icon from the home screen to create a new meeting.

She enters the meeting date, time, and duration as well as guest email addresses. She taps the submit button and, if all fields are filled with valid information, the system will create the meeting in the database, set up guest accounts, and email guests.

She will then be returned to the home screen and the meeting should appear in a list. The list is only for meetings she is involved in and are sorted where the top item is the closest scheduled meeting. She can tap on the meeting to edit details or add new guests.

She will receive notifications that can be viewed from the home screen which show updates for when a guest fills out their questionnaire and unlocks the door.

# Technical Platform

- Flutter v3.3.6
  - Targeting iOS, Android, and Web
    - Minimum iOS Version: 11.0
    - Minimum Android Version: 4.4 KitKat
- Firebase Core v2.1.1
- Firebase Auth v4.1.0
  - Authentication Service
  - Minimum Android Version: 4.4 KitKat
- Cloud Firestore v4.0.3
  - Firebase Firestore interface
- Web Scraper v0.1.4
- Embedded system for security/door lock

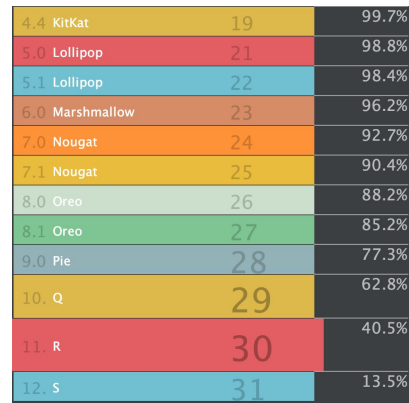
## iOS and iPadOS usage

As measured by devices that transacted on the App Store on May 31, 2022.

82% of all devices use iOS 15.

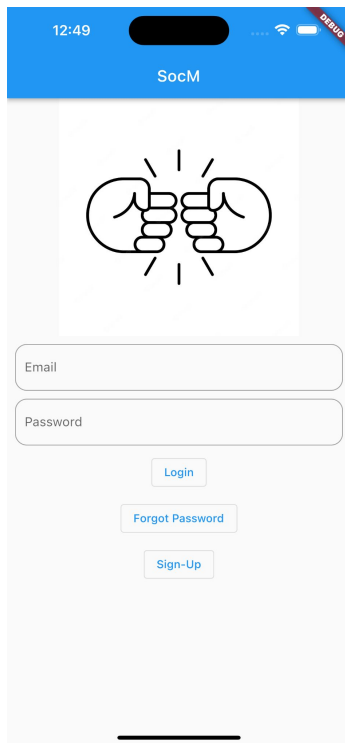


<https://developer.apple.com/support/app-store/>



Android Studio API Version Distribution Chart

# Software Available for Reuse



Authentication UI

```
Future<void> login() async {
  if (_formKey.currentState!.validate()) {
    try{
      UserCredential loginResponse = await _auth.signInWithEmailAndPassword(
        email: _email.text, password: _password.text);

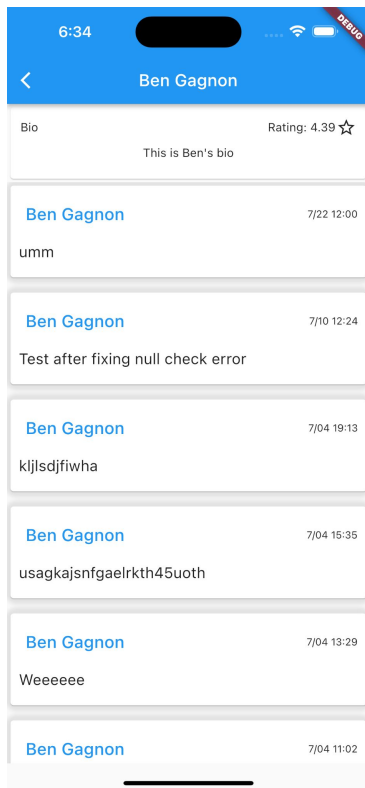
      if(loginResponse.user!.emailVerified) {
        Navigator.pushReplacement(
          context,
          MaterialPageRoute(builder: (BuildContext context) => const HomePage()));
      } else {
        snackBar(context, "User logged in but email is not verified");
        loginResponse.user!.sendEmailVerification();
        Navigator.pushReplacement(
          context,
          MaterialPageRoute(builder: (BuildContext context) => const HomePage()));
      }
    } catch(e) {
      snackBar(context, e.toString());
      loading = false;
    }
  }
}
```

Authentication Interface

```
validator: (value){
  if (value == null || value.isEmpty){
    return "Email cannot be empty";
  }
  if (!value.contains('@')) {
    return "Email is in wrong format";
  }
  return null;
},
```

Simple Email Form Validator

# Software Available for Reuse



Timeline with filtered and ordered posts

```
Expanded(child: StreamBuilder<List<Post>>(  
  stream: _fs.posts,  
  builder: (BuildContext context, AsyncSnapshot<List<Post>> snapshots) {  
    var posts = [];  
    var filteredPosts = [];  
    if (snapshots.hasError) {  
      return Center(child: Text(snapshots.error.toString()));  
    } else if (snapshots.hasData) {  
      posts = snapshots.data!;  
      for (var element in posts) {  
        if (element.id == widget.userProfile.id) {  
          filteredPosts.add(element);  
        }  
      }  
    }  
  }  
)  
  
  return filteredPosts.isEmpty ? const Center(child: Text("No Posts"),) :  
  ListView.builder(  
    itemCount: filteredPosts.length,  
    itemBuilder: (BuildContext context, int index) =>  
      PostCard(  
        filteredPosts[index],  
        Profile.fromJson(  
          filteredPosts[index].id!,  
          FirestoreService.profileMap[filteredPosts[index].id!]!  
        ).toJson()  
      ) // Profile.fromJson  
    ) // PostCard  
  ); // ListView.builder  
),  
) // StreamBuilder  
) // Expanded
```

StreamBuilder with filtered posts



The End :)