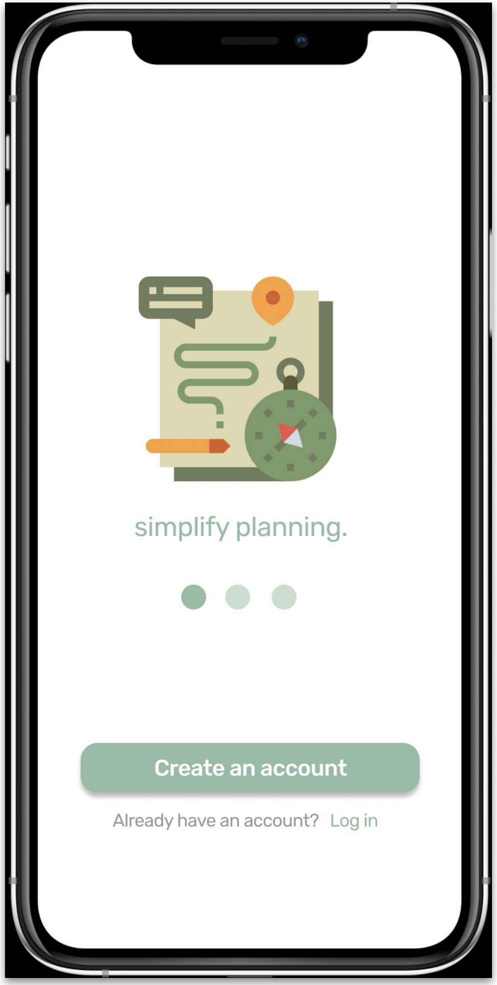
UNBURDEN

Catherine Poggioli - ISTE 454

What Is unBurden?

unBurden hopes to help you simplify planning, packing and camping. A wonderful experience can be ruined by leaving an essential item behind. unBurden will help you plan what necessities to pack and keep track of which have been packed.



Some Key Features

Login and Stored Data

- User Auth using Firesbase
- Stores User Data and Trip Data throughout sessions.

Packing Suggestions

- Get List of Suggested items to Pack
- Filter Suggestions by Categories like First Aid or Hygiene.

Plan Multiple Trips

- Users can create more than one trip.
- Users can easily navigate between trips.

Challenges to Simplify Later

The early implementation of Authentication meant that I was forced to learn how to structure pretty complex storyboards and passing of data and navigation. I had to learn how to use storyboard references.

I had to spend a lot of time figuring out how to read Firebase data and Incase I got to sharing trips, I wanted to make sure that the switch would be seamless. But, this meant for more complicated classes and data in the database.

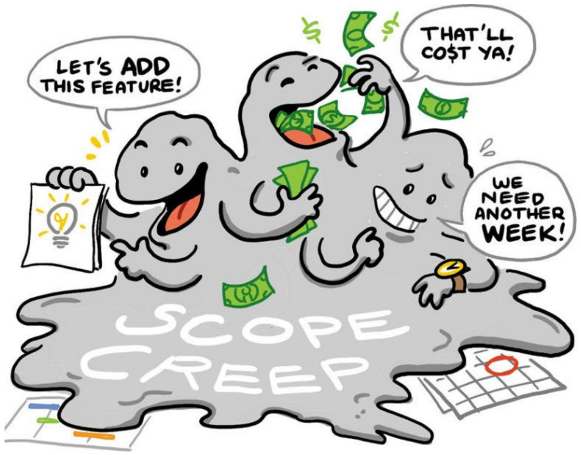
Challenges with Scope Creep

Expected Features

Create more than one trip
View Trip Details
Add items to pack
Mark items as packed
Items by category
Suggested Items by category

Final Features

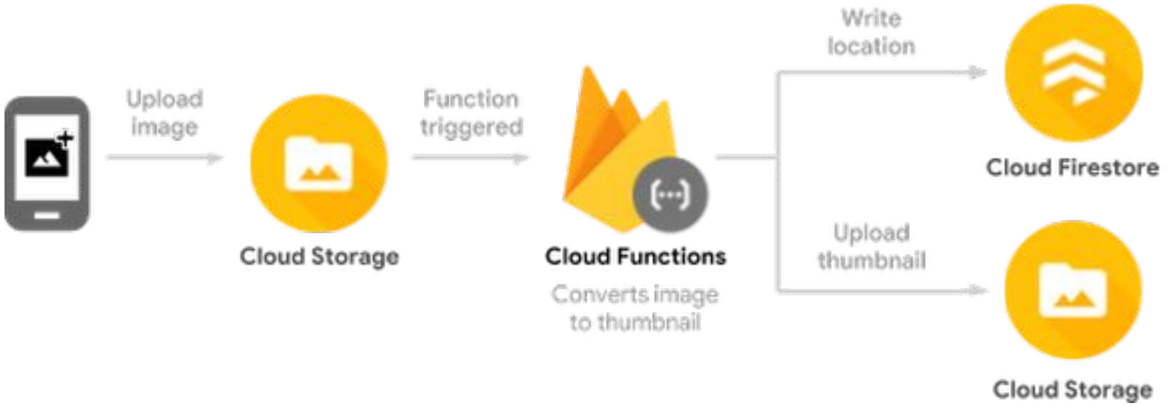
Sign up and Login
Authentication
Preservation
Ability to Create a Trip
Ability to Delete a Trip
Ability to Add Items to Pack
Ability to Mark Items as Packed
Suggestions Provided for Items to Pack
Filtered Suggestions
Stored User Data

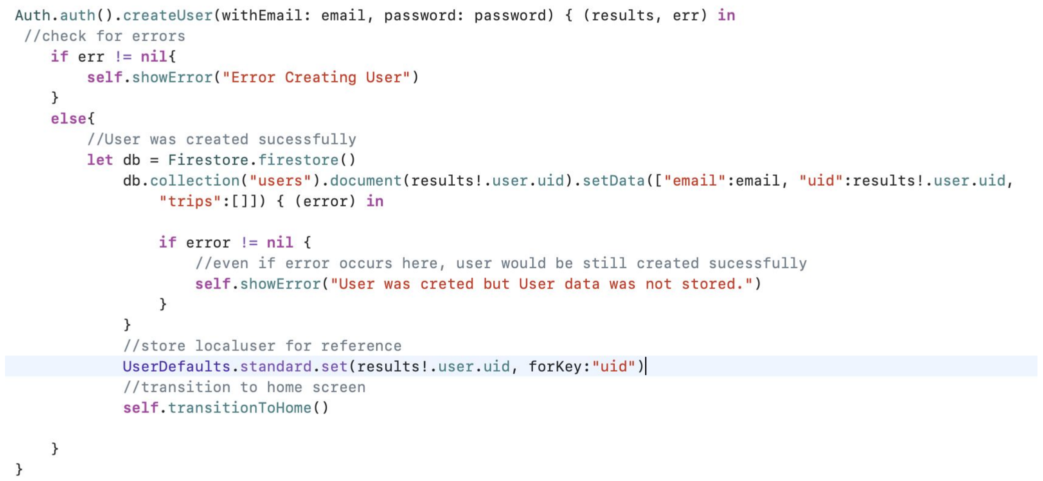


FireBase Is Great

There were some difficulties using Firebase documentation as there is a blur between the Cloud and Real Time Database Firebase.

The function call however were smoother than typical jquery api calls that I am used to.

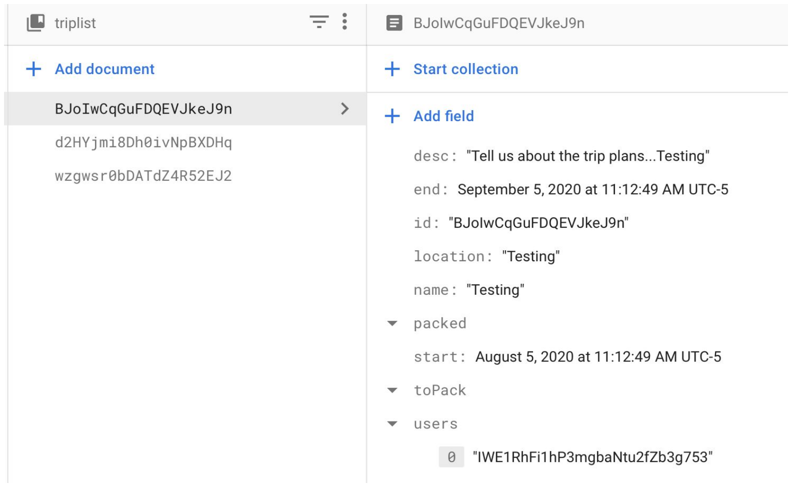




```
Auth.auth().createUser(withEmail: email, password: password) { (results, err) in
 //check for errors
    if err != nil{
        self.showError("Error Creating User")
    }
    else{
        //User was created sucessfully
        let db = Firestore.firestore()
            db.collection("users").document(results!.user.uid).setData(["email":email, "uid":results!.user.uid,
                "trips":[]]) { (error) in

                if error != nil {
                    //even if error occurs here, user would be still created sucessfully
                    self.showError("User was creted but User data was not stored.")
                }
            }
            //store localuser for reference
            UserDefaults.standard.set(results!.user.uid, forKey:"uid")
            //transition to home screen
            self.transitionToHome()

    }
}
```


triplist

Add document

BJoIwCqGuFDQEVJkeJ9n

d2HYjmi8Dh0ivNpBXDHq

wzgwsr0bDATdZ4R52EJ2

BJoIwCqGuFDQEVJkeJ9n

Start collection

Add field

```
desc: "Tell us about the trip plans...Testing"
end: September 5, 2020 at 11:12:49 AM UTC-5
id: "BJoIwCqGuFDQEVJkeJ9n"
location: "Testing"
name: "Testing"
packed
start: August 5, 2020 at 11:12:49 AM UTC-5
toPack
users
    0  "IWE1RhFi1hP3mgbaNtu2fZb3g753"
```

In order to not receive errors when calling and saving data to Firestore, I had to plan out the way my triplist would be stored. And the way to identify which users would have access to what data.

YoutuBe, Practice Exercises, GitHub

Youtube has tons of useful tutorials. I watched ones on everything from Data Modeling on Firebase to good design.

I found it extremely helpful to create multiple projects, I would test out features in separate projects and make sure they worked before trying to tackle the whole project.

GitHub was super useful during the development process. It is great being able to revisit working code if you broke something and weren’t sure why.

Questions?