# MealUp - Meal Scheduling App

**Project Leader:** Andrew Zeng
Andrew Zeng (andrewz@princeton.edu)
Chi Yu (cfyu@princeton.edu)
Annie Zou (azou@princeton.edu)
May Jiang (myjiang@princeton.edu)

## 1. Overview

*MealUp* is a mobile app that helps Princeton students schedule meals with their friends. Each user inputs their availability for meals over the next two weeks and then the app will help schedule meals between users through a request system. Users can easily keep track of who they plan to eat with for the week and easily request meals with friends through a user-friendly interface, and the app will find a time that works for both by matching the available times of the users. The app will support Facebook login and import Facebook friends for the user's convenience.

## 2. Requirements and Target Audience

### Problem

Currently students have to manually coordinate times to get meals with each other through messages. Many times friends say to each other "let's get a meal some time," but often it's too difficult to coordinate a time that work for both to meet, so the friends end up not getting a meal together. Our app aims to streamline the meal scheduling process and make it easy to keep up with your friends and eat with different people every meal.

### Intended Users

Princeton Students - Students with friends they already would schedule meals with and that tend to schedule meals in advance

- Save time and unnecessary messaging to schedule a meal
- Keep up with friends who a student may not see that often
- Keep track of which meals and which friends a user will eat with for the next two weeks

## 3. Functionality

Features

- Login using Facebook account to import friends
- Easy-to-use view of meals schedule for the week
- Request page to see sent and received meal requests
- Allows user to enter in times that he/she is available for a meal

- Potential features: group meals, repeating meals, push notifications

**Use Case 1:** requesting to get a meal with a specific friend/group of friends
A and B are sophomores who have been meaning to get a meal with each other for weeks now but have never been able to find a time that works for both of them. Every time A suggests his upcoming free time to B, B is unavailable and suggests her free time to A, which he can't do. They say they'll figure something out next week but are just faced with the same problem. They know they could potentially compare their entire schedules, but neither is comfortable asking for the other's entire schedule, nor do they want to take the time to do so every time they want a meal. They want to be able to quickly and easily find a time that works for both of them. With MealUp, A proceeds to the Friends tab (or Requests>Friends) and sends a meal request to B. MealUp only shows time options when both users are available, so A picks a time that he knows works for B, and they finally have their coveted meal together!

**Use Case 2:** find a friend who is available to get a meal at a specific time
C is a freshman who has a haphazard class schedule and is only able to get lunch while it feels like everyone they know has class, but they feel lonely eating alone. They don't quite feel confident directly messaging a friend that they may not know too well about their availability, but they are willing to get to know them better somehow (meeting people is what freshman year is for, right?). Even if they were to directly message this friend, it is very likely that they are unavailable when C wants to get lunch. C would like some way of immediately knowing all the people they know that are available during their lunch break. With MealUp, C proceeds to the Requests tab and selects "Request by Time" (or selects an empty slot on the Home Page). MealUp shows a list of Facebook friends that are available at the specified time(s), so C requests a meal with one of them, and they can get lunch together!

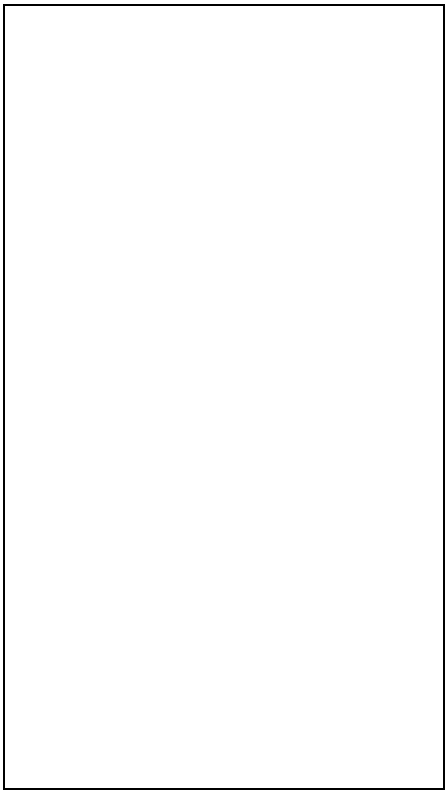
## 4. Design

1. User Interface

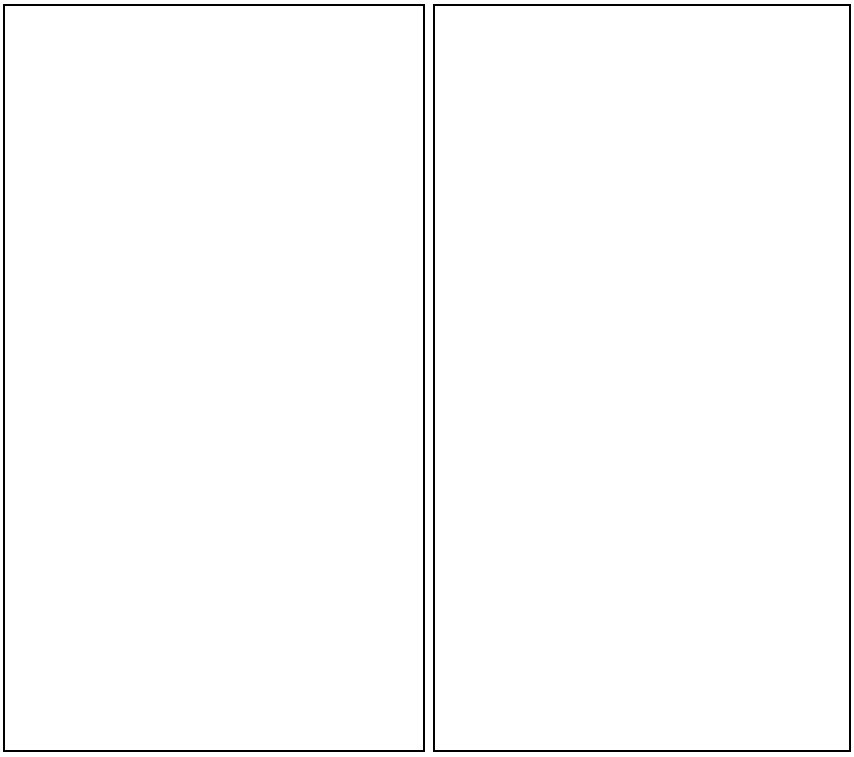
   First visit: Connect to Facebook & input initial schedule (free time)

   Various Tabs to navigate to once "logged in":

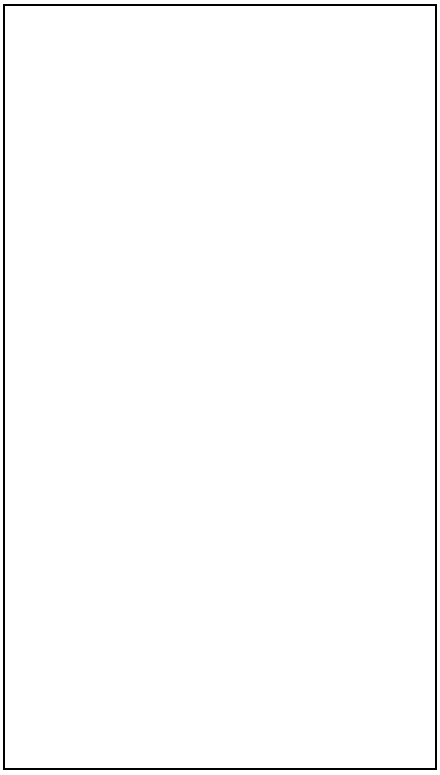
   a. Home page: Shows your planned meals



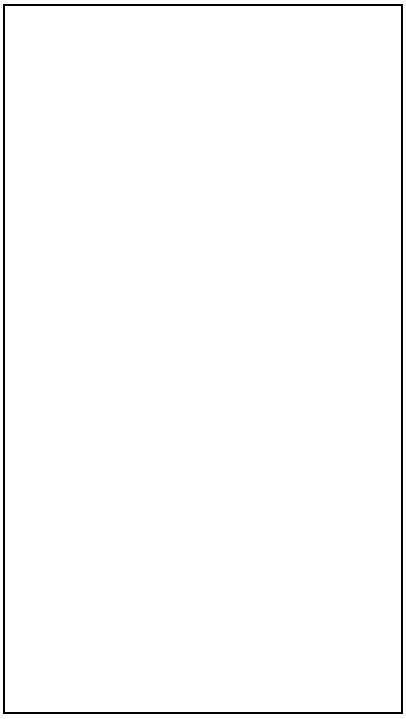
b. Request a meal *by time* or *by friend*



c. Friends list: you can favorite friends + also ranks friends by frequency; you can request a meal *by friend*



d. “Free Time” -- shows your weekly schedule and allows you to edit your available times for meals. Each week it defaults to the previous week’s schedule but you are prompted to make edits
    i. Horizontal scroll to view each day
    ii. Each day shows times as 30-min blocks like in WhenIsGood
    iii. Should be prompted to update it each week



Other Features (if any):

If we have time we want to implement push notifications so that users can be reminded when they have a meal that they scheduled coming up, or when they receive a new request for a meal, or when a request they sent out has been accepted and they have a new meal scheduled.

2. Process
    a. Authentication: we plan to use Firebase Authentication to implement the Facebook Login
    b. To get friends list: Facebook Graph API
    c. To find matches (for time and friends) and send update/notification when request has been sent
        i. Firebase Cloud Functions - no need to set up a server
            1. Making Request - showing proper results
                a. Request by Friend

i. Showing matching available times for user and selected friend
ii. Send Request (sender, receiver, date, time, location)

b. Request by Time
i. Show your available times
ii. Showing available friends at selected time
iii. Send Request (sender, receiver, date, time, location)

3. Data Management
a. We will be using Google Firebase Realtime Database or Cloud Firestore to store user data
b. For each user we will store:
i. List of available times (for each day of the week, time ranges)
ii. List of requests (sent and received)
iii. List of friends
iv. List of planned meals
1. For each meal we include time, location, and participants

## 5. Timeline

**Week 1 (3/17 - 3/24 [*spring break*]):**
Goals: do all the necessary learning, get a better idea of how our app will be created
Tasks:

- Learn React-Native (JS, React), Redux, How NoSQL works, Firebase Authentication, Database, and Cloud Functions, CSS Flexbox
- Set up project repository on GitHub
- Learn how to get list of friends from Facebook API
- Create project website

**Week 2 (3/25 - 3/31): Weekly TA meetings start this week**
Goals: Rough UI, figuring out how to read and write to our database, figure out basic idea of the Cloud Functions we need
Tasks:

- Facebook Authentication, list of friends done
- Have all the components we need figured out (possibly coded)
- Plan how to use Redux (organize state)

**Week 3 (4/1 - 4/7):**
Goals: rough UI completely coded, code database interaction code, Cloud functions
Tasks:

- Components coded
- Basic request implemented and received by other user
- Database code done

**Week 4 (4/8 - 4/14): Project prototype on 4/13**

Goals: make UI look better, bring all the parts together, test on ourselves

Tasks:

- Make accounts and test functionality between our accounts
- UI styling
- Cloud Functions done

**Week 5 (4/15 - 4/21):**

Goals: Implement potential features

Tasks:

- Push notifications
- Group meal scheduling
- Repeated meals

**Week 6 (4/22 - 4/28):**

Goals: Alpha test on 4/27

Tasks:

- Thoroughly test application with corner cases and debug

**Week 7 (4/29 - 5/5): Last class on 5/3**

Goals: Beta testing

Tasks:

- Continue to test and debug
- Test on real users

**6. Risks and Outcomes**

<u>Learning JavaScript and React Native:</u>

- We expect there to be a learning curve for JavaScript and React Native. React Native is a relatively recent framework so there might be some unresolved issues and new updates are still sometimes being released. If we run into any issues we can keep up with the discussion forum at discuss.reactjs.org and read up on Stack Overflow.

<u>Deploying to Android and iOS:</u>

- Even though React Native is a cross platform framework, we expect there to be some discrepancies between Android and iOS such as the way different components are rendered or appear, and the way that push notifications and other platform-dependent features work, and at the end to export the app we might have to go through Android Studio and XCode. We can work through these discrepancies as they arise and keep separate versions of the code for iOS and Android that take care of those discrepancies.

<u>Dependence on user to enter in data:</u>

- The app depends on the user to enter in their availability times so that the app can match user availabilities. This means the user has to enter the times that he or she is available over the next two weeks. If they don't enter in times or if they don't enter in enough times it can be difficult to match users and schedule meals.

Dependence on Facebook to connect to friends:

- The user must have a Facebook account in order to be able to use the app. We also have to filter list of Facebook friends to display the ones who have an account with our app.