**How to Use this Template**

1. Make a copy [ File → Make a copy... ]
2. Rename this file: "**Capstone_Stage1**"
3. Replace the text ingreen

**Submission Instructions**

1. After you've completed all the sections, download this document as a PDF [ File → Download as PDF ]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"

---



**GitHub Username**:DevB007

# IMalive

## Description

IMalive lets the people you care about know that you're alive.

Schedule specific messages to specific people on specific days at specific times. When the time comes, IMalive will send you a notification asking if you are indeed alive. When you confirm your aliveness, the app will send your message to the recipient.

# Intended User

The intended user for Still Alive is anyone who wants to let people who worry about them know they are okay. For example, students who just left for college who have moms that message them every day and ask if they are okay.

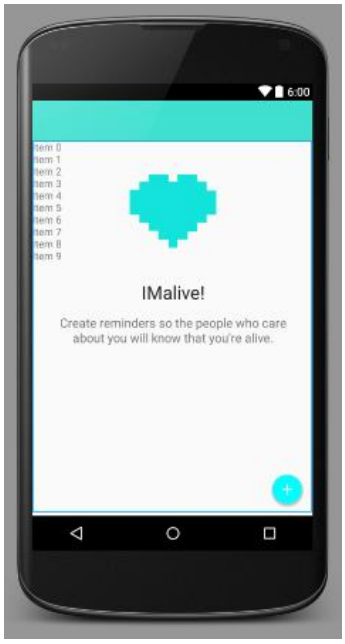
# Features

List the main features of your app. For example:

- Pick a contact from your phone's address book
- Configure days and time for a specified message to be sent to the selected contact
- Easily manage existing configured messages
    - Toggle on and off scheduled messages
    - Update configuration or delete entirely

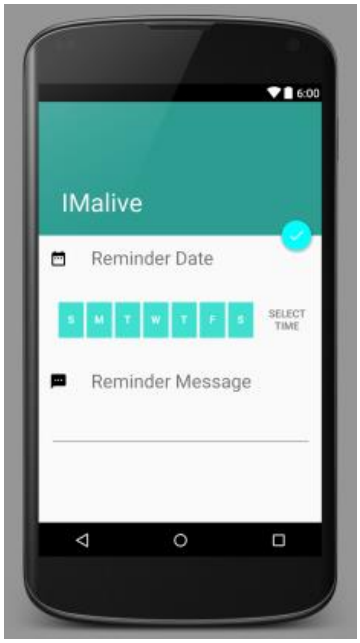
# User Interface Mocks

### Main Screen - Messages Dashboard




The main screen is a dashboard where users can manage existing messages they have configured, as well as create and configure new messages.

### Message Composer Screen




The message composer screen lets users specify a date and time as well as a message to be delivered to the selected contact.

## Key Considerations

### How will your app handle data persistence?

The app will use a 3rd-party library to persist simple data.

### Describe any corner cases in the UX.

Since the app will use alarms for scheduling push notifications, the alarms will need to be rescheduled if the user restarts their device.

### Describe any libraries you'll be using and share your reasoning for including them.

I'll be using Realm.io for the database to store and retrieve data created within the app. I choose Realm because it makes implementing a SQLite database fast and easy.

# Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and decompose them into tangible technical tasks that you can complete incrementally until you have a finished app.

## Task 1: Project Setup

Before I begin developing my app, I'll first need to add Realm as a dependency in my app's build.gradle file. Once I have Realm available, I'll be able to save and retrieve the data users will be creating in the app.

## Task 2: Implement UI for Each Activity and Fragment

- Build UI for MainActivity
    - Implement FloatingActionButton which starts an intent to the default contacts picker when clicked
    - Implement ListView that displays the scheduled messages users created via the MessageComposerActivity
- Build UI for MessageComposerActivity
    - Implement UI control to select one or many days of the week
    - Implement UI control to select a time
    - Implement EditTextView for the user to type a message
    - Schedule alarms with AlarmManager for the date set by the user so a push notification can be sent

## Task 3: Handle Corner Case

If the device is restarted, the scheduled alarms must be rescheduled so that users can be notified via push notification to confirm their aliveness. I will register a receiver in the manifest to listen for when the phone is rebooted to achieve this.

---

**Submission Instructions**

1. After you've completed all the sections, download this document as a PDF [ File → Download as PDF ]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"