University of Arkansas – CSCE Department
Capstone I – Final Proposal – Fall 2021

MAGIC: A Transportation Toolkit and Application for Older Adults

Emily Lea, Patrick Page, Patrick Karangwa, Alan Torres, Sailesh Sirigineedi

Abstract

Among the growing population of older adults there is a lack of reliable, familiar transportation services to get them to the grocery store, medical appointments, etc. in order to meet their primary needs and even for non-essential tasks such as recreational activities or meeting with friends and family. This could lead these older adults to seek unfamiliar and confusing transportation services. The main objective is to build a mobile application with a toolkit that will assist the elderly in locating appropriate, safe, and reliable transportation services in the area, as well as scheduling a ride if a personal driver is available. After this, the next-most important aspect will be designing the application to be as user-friendly as possible to accommodate the elderly. To build this easy-to-use mobile application, we will be using HTML, CSSS, React Native, Python/Django, and PostgreSQL for most of this project's needs.

1.0 Problem

In the US, the population of older adults ages 65 and over is expected to quickly rise within the next decade. By 2030, roughly 20% of the population in the US will consist of older adults [1]. Arkansas is currently within the top 50th percentile in the nation with regard to the growth rate of the aging population [2]. Rural areas, many of which have poor, sparse transportation resources will see a greater increase in their older adult population than other regions will [1]. Conversely in more densely populated urban areas with fewer numbers of older adults, public transportation may be one of the first resources to be cut in times of budget issues.

Transportation by means of an older adult driving themselves is not possible in many cases due to poor health or physical mobility issues. Transportation by means of family members may be infeasible if the older adult does not live near relatives or if a relative is unable to transport them on specific dates and times. Older adults in areas that do have public transportation may be unfamiliar with using it, may be concerned for their health and safety in using it, or may be unaware of the transportation resources available to them.

In all of these situations, the core issue is a rapidly growing demographic's lack of information about safe, reliable transportation resources in their region. Even if transportation resources are present in many regions, older adults may not be aware of the transportation options they have. Essentially, not knowing about transportation resources translates to lack of transportation. Lack of transportation hinders the ability of the older adult population to care for their basic needs

such as getting to the grocery store or to medical appointments. This can also interfere with their ability to meet secondary needs such as going to recreational activities or socializing with friends and family. If the growing population of older is adults is left with little resources regarding access to transportation resources, they will be unable to meet their basic needs or may be forced to seek out unsafe, unfamiliar transportation methods.

2.0 Objective

The objective of this project is to create a user-friendly mobile application to assist the elderly in finding safe, reliable means of transportation to medical appointments, grocery stores, recreational activities, and a variety of other necessary destinations. The mobile application will include a toolkit, very easy-to-use for the elderly, which allows them to locate the appropriate transportation service or personal driver in their area, and schedule a ride to whichever appointment they need to get to. The MAGIC toolkit app is designed with the elderly in mind because the primary purpose of the app is to help the elderly, many of whom are not very technologically inclined. With this in mind, the MAGIC Toolkit application will also attempt to keep the elderly clients as comfortable and care-free as possible during this process, placing an emphasis on drivers they can trust, mostly friends and family.

When the common word user-friendly is used in this context, it means that we have to ensure the objective of our application will be met in its entirety. We will be taking into account that if an elderly client attempting to use our app cannot figure out how to use it, then the application is pointless to them and does not achieve its ultimate purpose for that particular client. Therefore, ease-of-use is one of our primary objectives, only surpassed by the element which assists the clients in getting the proper transportation at the correct time. This objective involves designing our application so that even the least technologically inclined clients will be able to use the application to achieve their goal of getting reliable transportation when they need it most.

To be more specific, we will use React Native, Python/Django, and a PostgreSQL database to implement a mobile application with a variety of different location-based transportation resources to assist the elderly in getting to and from the places they need to go. We intend to make the application easy for anyone to use, regardless of their background/previous experiences with different technologies. With this application, the elderly will be able to receive assistance in an area that might have been previously lacking in their lives. We strive to place an emphasis on facilitating friends and family volunteers to give their elderly friend/family member a ride as opposed to complete strangers. We emphasize this because studies have shown that the elderly are weary of getting transportation assistance from complete strangers. Our objective will need to be met with the general idea that transportation services from friends and family come first, and then other transportation resources will be made available if that is not available. We want to provide a safe, reliable transportation assistance application which leaves the client feeling care-free and as comfortable as possible every step of the way.

3.0 Background

3.1 Key Concepts

For this project, there will be users on both iOS and Android devices, because users will have to have a smart phone to work this application and somewhere around 99.3% of all smart phones are run by either Android or iOS operating systems. This is important because this lead us to our design decision of what programming languages and frameworks to use when developing this application.

We decided to use React Native, a very versatile JavaScript framework, which has gained its popularity due to the high reusability of code. In other words, this is a great framework for cross-platform development, as most of the code will work on either iOS or Android devices. It is approximated that the reusability of code between Android and iOS devices is 90-95%. This is why we decided to implement the front-end of our application with React Native.

Our team also decided to use Django, a web framework based on the Python programming language. Django was designed to help developers get from their initial design concepts to project completion as quick as possible. It follows the typical model-template-views architectural pattern. Django is fast and easily scalable. These are just some of the reasons we decided to use Django to implement the backend of our application.

For our database purposes, we will be using PostgreSQL, an open-source database management system. This is where we will be hosting all of the usernames, passwords, driver information, client information, home addresses, and any other important information needed to help our application successfully run.

3.2 Related Work

Here are the services that are currently available and some of the shortcomings that our application will improve upon:

- **Uber:** The company Uber has launched several programs to assist senior citizens get to their destinations safely:
    - Uber launched a program in Gainsville, FL that aims at teaching seniors how to use their Uber app [3].
    - *UberWAV* was launched in Toronto, Canada to give people with wheelchairs an option to choose a wheelchair-accessible vehicle [3].
    - *UberASSIST* was designed for people with special accommodations [3].
        - This service provides trained drivers to help elderly adults.
        - Vehicles are wheelchair-accessible.

  The shortcomings that will be addressed by MAGIC:

    - Senior citizens have reported that the user interface of the *Uber* app is confusing and makes the app hard to use. MAGIC has a simple user interface that is more friendly to older adults.
    - MAGIC focuses on family, friends, and neighbors being the ones offering rides instead of complete strangers, an issue that has raised concerns among older adults.
- **Go Go Grandparent:** this program is specifically oriented towards senior citizens and it accomplishes the following:
    - The user purchases a membership for $9.99/month [4].
    - The subscription allows them to call a number which prompts them to make choices on the dial pad, and then an Uber or Lyft is called on their behalf, without having to create an account for any of those platforms [4].

    - The user can dial back the number and select to be dropped off home once they are ready to go back home.

  The shortcomings of this service that will be solved by MAGIC include:

    - Price: this service is very expensive because on top of the $9.99/month subscription fee, the user has to pay $0.27/minute fee + the base fare for the Uber or Lyft. MAGIC will be free-of-charge because it will be based on volunteer drivers [4].
    - This option is not convenient for seniors who suffer from hearing loss since it requires them to listen to prompts on the phone. MAGIC will encompass a very simple and straight-forward user interface.
- **Lyft:** The company Lyft has recently added a feature that would allow senior citizens to order a ride by phone call. Here is how they accomplish that:
    - The user calls a phone number and then they get to choose various prompts to connect them to a driver, without the necessity of having the mobile app [5].
    - Updates are communicated via text message [5].

  The shortcomings that will be addressed by MAGIC are:

    - More simplicity offered by MAGIC compared to having to dial and hear different prompts, which can be very confusing.
    - The application would cost no money.
    - MAGIC focusses on family, friends, and neighbor's volunteer services, eliminating the issues that can come with riding with strangers.

4.0 Design

4.1 Requirements, Use Cases, and Design Goals

The basic features and requirements include providing information about available transportation resources as part of a toolkit, providing trip planning guidance based on origin and destination, and trip coordination for older adults with families and friends. We would also like to have trip coordination with volunteers. We want account creation and logging in for all users – both older adults and drivers – as well as a system for older adults to rate the drivers. The following requirements and use cases will be included in our design process:

Requirements:

- Provide reliable trip planning guidance and facilitation for older adults, i.e., passengers.
- Must be user-friendly to target the elderly population with little to no technical skills/experience.
- Must have both a passenger application and a driver application.
- Must allow drivers to set and update the times they are available to provide transportation.
- Should allow drivers to send a friend request to a passenger by looking up their name in the database.
- Passengers should be able to accept or decline requests from drivers to be included among the trusted drivers.
- Provide the ability for the passenger to view transportation options from only friends, family, or other trusted individuals in their social circle.

- Provide handicapped accessible transportation resources for passengers in need of these resources.
- Provide a rating system for passengers to rate their drivers, an important part of many transportation-related applications.
- Must provide the ability for passengers to schedule a date and time for a transportation request.
- Provide a list of available drivers from which the passenger can choose who they would prefer to receive transportation from.
- Must have push notifications for drivers to receive with transportation requests in their area during their available time slot.
- Must have push notifications for passengers to receive, with their request being accepted/denied by the driver they requested.
- Provide the ability for passengers and drivers to create an account to log into application.
- Must have a navigation menu to accompany any buttons on the home screen and other views within the application.

Use Cases:

- Elderly adult schedules a trip some amount of time in advance, producing a list of drivers available at that date and time from which the passenger, i.e., the client, can pick whichever they would like. The requested driver received the request and accepts it, then gives the client the requested transportation at the given time.
- Elderly adult schedules date and time of requested ride, selects preferred driver, but driver declines request for whatever reason. The elderly adult is notified of the driver's denial of their request and is able to send a request to one of the other available drivers.
- Driver accepts request to give transportation to a client at a scheduled date and time, but has to cancel the ride at the last minute due to emergency or for whatever reason. Must at least consider some way to quickly fix this to ensure the client still gets a ride if at all possible.
- Client requests a ride right when they need it, without giving any advanced notice. This would require some consideration into a prompt response system or even a category of drivers which are available for quick rides, or some other design and implementation to handle this use case. If this use case is not considered during design and implementation, the client would quite likely have to wait a good amount of time for a driver to accept the transportation request.
- Client requests a ride from a specific place, but then leaves the location where they requested the ride from. Need to consider real-time geo-location services (when permissions are enabled) to allow the driver to find the client and/or ability for driver to call the client on the phone.
- Driver arrives to pick client up at scheduled time but the client is not there. Driver might need a way to call client to let them know that they are ready to give them transportation.

4.2 High Level Architecture

1. Architecture

MAGIC will consist of an application for trip planning as well as a toolkit for transportation resources which can be used by older adults to meet their transportation needs. This application and toolkit will be subdivided into two parts: one side of the application will be designated for

older adults, i.e., the passengers, and the other will be for drivers who will be completing trips. Since this application will be supported on both iOS and Android mobile devices, there won't be a need for developing separate applications for both of those operating systems thanks to React Native.

To create an account, a user will first need to choose whether they are registering as a driver or as a passenger. Passengers will enter the following information to sign up: first and last name, home address, email address, photo (optional), phone number, and password. Drivers signing up will need to enter the following: first and last name, phone number, email address, photo(required), vehicle's license plate number, make and model of vehicle, if the vehicle is wheelchair accessible, availability schedule, and password.

To log into their accounts, both drivers and passengers will use their phone number and the password they chose in the sign-up process. The password will not have the requirement of a certain number of characters or the exclusion of certain kinds of characters. We will include the ability for a person to reset their password in the application.

The passenger will be able to plan a trip by entering their destination and optionally the reason for their trip, whether it's for buying groceries, medical appointment, etc. The passenger will also have the option to schedule a trip in the future. This will be done using a calendar API from Google Calendar or Apple Calendar, which would also toggle an alert when their scheduled trip is upcoming. On top of trip planning, the passenger will be able to choose from available drivers in drivers in two ways. First, if they need a ride immediately, they will be able to see a list of available drivers (family and friends who have registered), choose whoever they desire to give them a ride. Second, if the passenger wants to schedule a ride in the future, they will be able to see the driver's availability that time and schedule with them. If there are no available drivers during that time, that's when they would use alternative transportation resources nearby using the toolkit.

On the driver's side, they will have the option to set their availability to "now" or provide a schedule for when they will be available. If the driver chooses to set their availability status to "now", they will be able to accept or decline incoming requests for a ride. Once a request comes in, the application will notify the driver about who requested a ride and also provide their location. Once the driver accepts, they will be routed to their preferred navigation application, which will provide directions to the passenger's location. Upon their arrival to the passenger, the driver will notify the passenger of their arrival. Once they pick up the passenger, the driver will be routed to the passenger's destination in the same way they were directed to the pickup location. If a request for a trip scheduled in the future comes in, the chosen driver will get notified and have the option to either accept or decline. When they accept, the trip will be scheduled on their calendar and an alert or reminder will remind the driver of the planned trip before the scheduled time.

On top of trip planning, this application will also provide an easy way to access alternative transportation resources available in the area. If the passenger can't find a driver during the time they request a ride, they will have an option to easily choose other transportation resources like public transit, wheelchair accessible vehicle services, etc. The app will have a compilation of all those resources arranged in an easily accessible format with a friendly user interface. There will be information such as phone numbers a passenger can call, bus routes, operation hours, and so on; all in one place.

- ***Components***

- Passenger/Older adult side
  - After downloading the application, the user will be prompted choose whether they are the driver or passenger. For new users, they will have to first choose which side of the application they will be using and then creating an account through there. Afterwards, they will have to just log in with their credentials and they will be redirected to their appropriate side of the application.
  - When signing up, they will be required to provide their first and last name, email address, home address, and phone number. Then, they will create an account username and password for authentication purposes.
  - *Home Page*: this will comprise of an easy and straight to the point user interface. It will have three main buttons, one to take the passenger home, another one to plan a trip to a desired destination, and another one to get access to alternative transportation resources. The home page will also have a navigation pane that allows the user to access various functionalities like managing notifications, reporting an issue, etc.
  - *Go Home:* when signing up, the passenger will be asked to provide their home address. This address will be stored in the database and used whenever the passenger wants to go home so they don't have to enter it all the time. Once they click on this button, they will be prompted to choose from available drivers who can give them a ride. The passenger will be able to filter through drivers. They will have three options: family and friend drivers, non-family and friend drivers, and all drivers. Once they make their choice, a notification will be sent to the driver letting them know that a passenger has requested a ride. Once the driver arrives, the passenger will receive a notification letting them know that the driver has arrived.

    Passengers will also have the ability to accept or decline incoming requests to be added to a list of "trusted" people. The request is made by the driver and once accepted, the driver will be classified among the friends and family.

  - *Plan a Trip:* upon clicking this button, the passenger will be prompted to enter the address of their destination and as they type, a drop-down menu will provide suggestions based on what they are typing. Once they choose a destination, they will get to choose whether this trip will be in the future or immediately. If they choose to plan a future trip, they will type the date and time of the trip and a list of available drivers will come up. A notification will be sent to the driver. If the driver accepts, both the driver and passenger will have the ride added to their phone calendar. If the passenger chooses to request a ride immediately, they will be prompted to choose from a list of available drivers, select one and then submit a request, along with their destination.
  - *Local Resources:* As mentioned before, this button will have a compilation of the available transportation resources as an alternative. When the passenger clicks on this button, they will see transportation alternatives available in their region. They should be able to click on any one of them to see a short summary of what the transportation resource

offers and explain how it operates. They should be able to also make a call to the transportation service by the clicking on the phone icon. This will open up their phone app and they can make a call to the transportation resource.

- *Navigation Pane:* this will have a Settings and help center button. Settings should include the ability for the passenger to update their home address as well as adjust their notification preferences. The help center button will direct the passenger to a form where they can report an issue they faced with the application. This will be sent in form of an email to the developers' email addresses where they can properly address it.
- *Account Management:* this functionality will be implemented through the user account icon. When the icon is clicked, the user should be able to access and manage account information such as their phone number and home address. When clicked, users should also be able to logout of their account and upload or change their profile picture.

- Driver's side
  - Drivers will download the same application as passengers do and choose the option for drivers. Just like passengers, they will be asked to login if they already have an account or sign up if they don't.
  - The driver's sign-up page will be a bit different from that of a passenger. On top of providing their credentials and address information, they will be asked to enter information about the vehicle they will be using. In addition to that, all drivers will be required to provide a profile picture, that way their passengers can identify them and avoid confusion.
  - By default, all drivers will appear in an unfiltered list of available drivers to the passenger. The driver can send a request to the passenger to be added to a list of trusted people.
  - *Home Page:* after logging in, the user interface of the driver's side of the application will be much similar to the one for the passenger. The home page will have three buttons, one that allows the driver to set their availability to "now", another one that allows them to provide a schedule for when they will be available in the future, and another one that allows them to send a request to the passenger.
  - *Available To Drive:* this button, when clicked, will allow the driver to indicate that they are readily available to offer rides. They will select how long they will be available. That will automatically make them available for when a passenger associated with them requests a ride. When available, the driver can receive requests from passengers and be able to accept or decline the requests. The request will come in form of a notification.
  - *Set Availability:* this is a button that lets the driver provide a schedule for when they are available to drive in the future. This functionality will be much similar to that on the passenger's side where they select a date and time that they will be able to offer rides. Once they choose their availability, it will show up on their phone calendar. The calendar in their phone will send them a reminder when the trip is approaching.
  - *Send Friend Request:* when clicked, this button will allow the driver to search a passenger by their name and/or phone number in the database so

they can be added to a list of trusted drivers. When the request is accepted, the driver will be added to passenger's list of trusted people.
- *Navigation Pane:* drivers will have a navigation pane with app settings and a way to report an issue. They should also be able to manage notifications.
- *Check in to location*: drivers will have the ability to check in on the app once they arrive at the older adult's location. This will send the older adult a notification that their ride has arrived.
- *Account Management:* just like on the passenger side, drivers will be able to access their account information through clicking the user account icon. When clicked, the user can be able to change their profile picture, vehicle information, as well as their address. They should also be able to logout from their account as well.

2. Technologies Used

- This application will use PostgreSQL as a database management system to store and manage data that is collected at sign-up or during app usage.
- The frontend of the application will be implemented using HTML, CSS, JavaScript, and React Native as a framework. React Native is supported on both iOS and Android operating systems, and therefore won't require making two applications for both systems.
- Python will be used on the backend side of the application, using Django as a framework.
- We will use Git to make our repositories as well as to coordinate and integrate each of our portions of work with one another.
- Trello will be used to manage the project as well as to document and track the progress on our tasks.
- Testing can be done using a combination of physical devices and simulators such as BrowserStack.

3. Interface Design with Screenshots

The designs displayed below show an intuitive, user-friendly interface. Many of the buttons have an icon indicating what they are for. The text is large and easy-to-read, and contrast exists between the text and the background. The color palette chosen helps distinguish different parts of the application without being harsh or difficult to decipher. The Home pages for both the driver and passenger display three buttons main buttons and a "help" button, each with icons illustrating what they represent. In the trip planning process, each stage has an explanation of how to use the components. The Resources page contains instructions for how to view descriptions of the resources, and it contains phone icons to call each of the resources from within the application.

Figure 1: Home Page for the passenger side of the application.

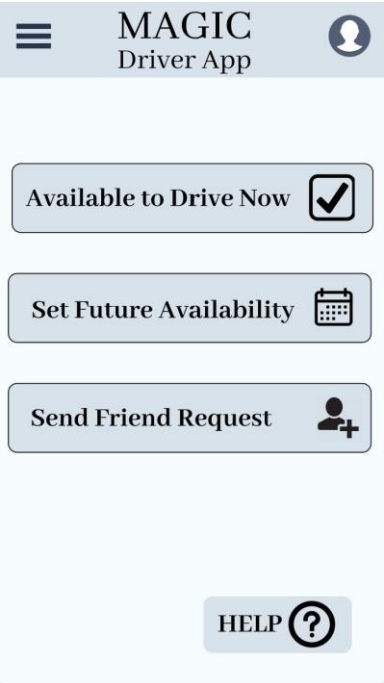

Figure 2: Home Page for the driver's side of the application.

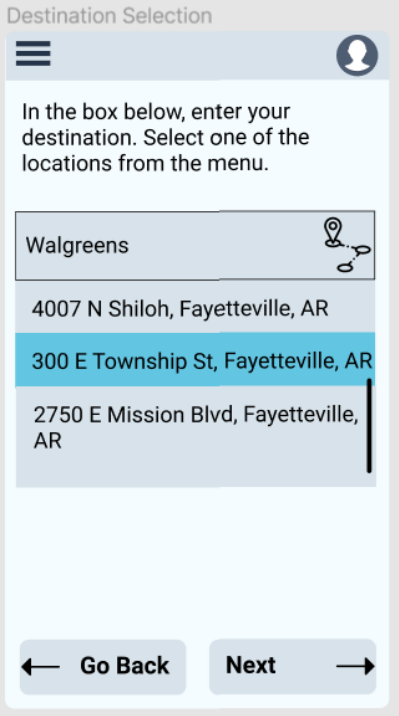

Figure 3: Destination Selection for the passenger.

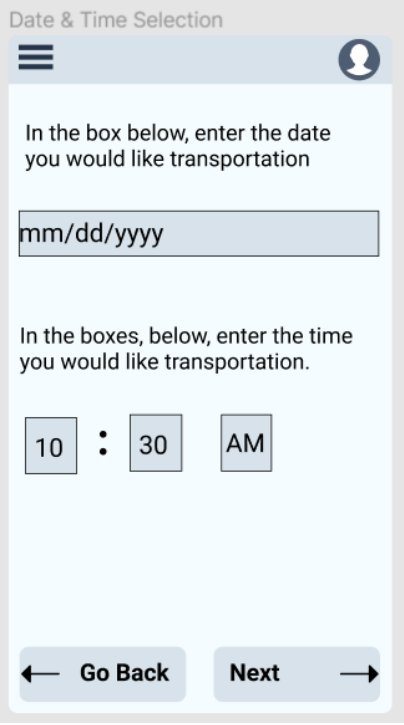

Figure 4: Date and time selection for the passenger.

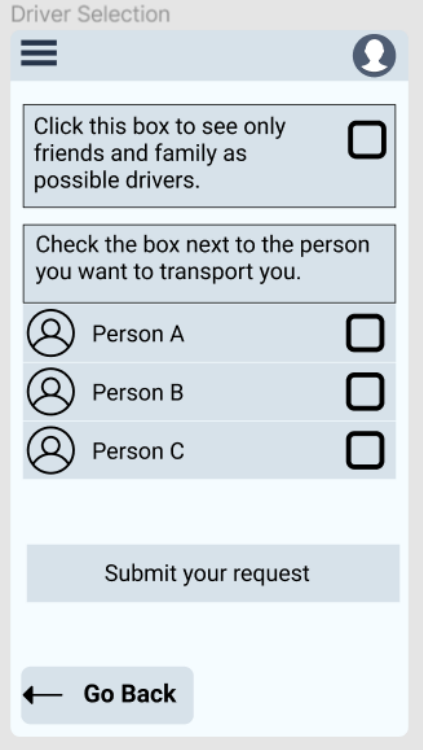

Figure 5: The passenger chooses the driver to give them a ride.

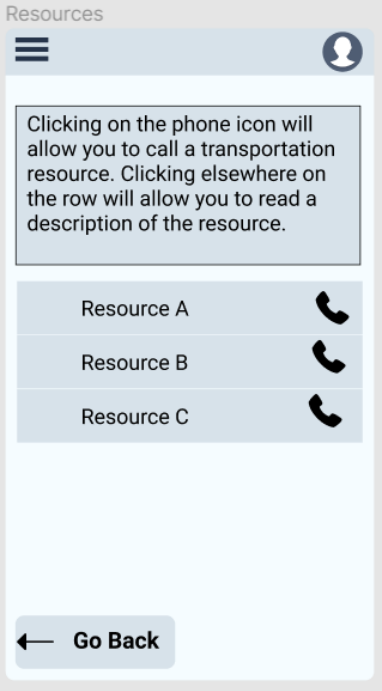

Figure 6: The Resources page

4. Implementation

The general flow of the application development can be classified into the following stages: database, account creation and login, home page and shared layout elements, friend requests, trip

planning, resources, driver rating system. Each of the five teammates will be assigned different tasks to work on in parallel over the course of one-week sprints. Larger tasks may be divided up among two people.

After setting up our development environments and repositories, the database will be created. The database will store all the users' - both drivers' and passengers' - information that they gave in the sign-up process. An ID will also be assigned to each user and stored in the database. Once the database is complete, the home page and shared layout elements will be added to the application.

Two home pages and navigation bars will be created: one for the driver account and one for the passenger account. We will work in parallel on this task by splitting it into the driver and passenger elements. Within each of the two subtasks, those can be split into parallel tasks as well. The home page and any pages that the navigation bar leads to can be worked on at the same time by different people. Because the home page consists of buttons, either Pressable or Button components in React Native can be used. The styling applied to one button can be stored into a shared styling file and applied to other buttons, as well. Styling applied to other elements that will be consistent across the application can be saved in a shared file and applied to other elements later on.

Once the home pages are complete, we will work on the sending/receiving and confirming/denying friend requests. Friend requests are a request sent by a driver to an older adult asking to be considered as a trusted driver. If the older adult accepts the request, the driver will be added to a list of drivers considered to be trusted. If the older adult denies the request, the driver will not be added to the older adult's list of trusted drivers. This task, once again, can be broken up into passenger and driver elements. On the driver's application, this task requires a page containing text input search bars for entering a name and/or phone number by which to search for someone to send a friend request to. A list will appear below the search bar indicating results matching the user's input. Each item in the list will be a clickable element that, when clicked, asks the driver whether or not they would like to send a friend request to that person. If confirmed, a request will be sent to the older adult. If denied, no request is sent, and the driver is taken back to the search inputs. Another subtask will be testing whether or not the older adult receives a notification indicating that they have received a friend request. Once testing is completed, the next parts of the task are to enable the older adult to confirm or deny the friend request and to send a confirmation or denial notification to the driver. Once again, testing will need to be done to ensure the driver receives confirmation or denial of the request. Another subtask that will be worked on in parallel with the passenger elements is saving a confirmed driver to a list of trusted drivers for that passenger.

The next stage of development will be to implement trip planning. The tasks for this stage will be worked on in parallel by dividing it up in the following way: geolocation, location finding and selecting, scheduling, driver selection, and sending/receiving notifications. For geolocation, one of a few possible React Native packages will be used to determine the older adult's starting point in planning a trip. This location will be saved to the application. For location finding and selecting, a text input box with a dropdown list of nearby locations will be implemented. The nearby locations will be retrieved using a tool like Android Places and iOS Places SDK. Each list item will be selectable. Selecting a location from the list will designate it as the location the passenger wants to be transported to. For scheduling, a date input box and a time input box will be implemented to gather the passenger's desired date and time of travel. The date and time will be saved in order to later display a list of drivers with availability schedules corresponding with

the older adult's desired time and date of travel. For driver selection, a checkbox will be implemented to allow the older adult to choose whether or not they would like to be shown only available drivers from their list of trusted drivers. Based on their selection, a scrollable list of available drivers will be displayed. Each list item will be selectable. Selecting an item will show test asking the older adult if they would like to send a transportation request to that driver. If confirmed, a request will be sent. If denied, the older adult will be taken back to the list of available drivers. A subtask will be implemented at this point to test the sending and receiving of transportation requests between the older adult and potential driver. Once the sending and receiving of transportation requests has been tested, confirmation/denial of a transportation request on the driver's side will be implemented. An alert will be implemented on the driver's side of the application displaying the details (time and date, requester, destination) and offering the options to confirm or deny the request. If a driver confirms the request, a confirmation response will be sent to the older adult; if a driver denies the request, a denial response will be sent to the older adult. Another subtask will be implemented here to test that the confirmation/denial responses from the driver are being sent to and displayed on the passenger's end. The final element of trip planning is as follows: if a transportation request is confirmed, the time, date, and destination of the request will be added to both the older adult's and the driver's Google or Apple calendar.

The functionality to plan a trip home has been mostly covered in the previous paragraph. The primary difference for this component, though, will be that no destination selector will be present. Because the older adult already entered their home address in the sign-up process, their home address was stored in the database and does not need to be entered again. The time/date and driver selectors will apply to this component, however. Another common element to any kind of trip planning will be a button the driver can use to "check in" to an older adult's location once they have arrived. When a driver checks in, the passenger will receive a notification that their ride is ready.

For the resources component, a page will be implemented that displays a list of transportation resources unique to someone's area. Each item in the list will be clickable. When an item is clicked, an explanation of the resource and how to use it will appear in a drop-down description as well as a button that will allow a user to call the transportation resource. For the scope of this project, we will hard code in local transportation resources, then test the component.

Once the resources component is complete, the final stage of development will be implementing an optional rating system for older adults to rate a driver. At the end of the older adult's transportation, they will be shown a popup asking if they would like to rate their driver. If a passenger clicks "yes", they will be able to rate their driver from one to five stars.

5. Lessons Learned

The lessons learned from last time is to include the necessary functionality for the drivers on our platform. This includes drivers' registration, job notifications, and job selection. We will plan to have our platform handle how drivers are notified and how they can accept and complete driving jobs on the front end as one part of user base of the platform. Also, we learned that account creation is very easy to implement using bootstrap modules/templates that are already available, so implementing account creation, registration, etc., will not be a problem.

6. Potential Impact

The potential impact of this project is to improve the quality of life of the elderly clients by assisting them with transportation, making their lives less stressful and more fulfilled. This will

have functionality on the front end for both halves of the user base, the elderly people who are needing rides, and the drivers who are able to provide the rides. We want the app to be user friendly for both of the user bases. This app will specifically be designed to be easy to use for old people as it will include a toolkit for the most common errands.

7. Future Work

Possible future work on this application includes background checks for the drivers as well as implementation of in-app navigation for the drivers to eliminate the need for 3rd party navigation apps. As for the elderly clients, future work includes adding languages other than English, suggesting activities in the area, the ability to add more transportation resources, and the ability to easily contact emergency services for both the driver and client application users.

4.3 Risks

Risk Summary	Mitigation Measures
An older may not know their exact current location when wanting to go home.	The older adult's location is saved through a geolocator that will be implemented with React Native.
The older adult may be unfamiliar with mobile applications, specifically with ones regarding transportation	The user interface will be made simple and intuitive, and instructions will be present in trip planning to resolve any confusion that may arise.
The older adult may not trust just anyone to give them a ride to a given location.	The older adult will have the ability to build a list of friends and family by accepting a friend-or-family request from someone. They will have the option in trip planning to view only the drivers listed as friends or family.
A login process needs to be present to mitigate security risks, but needs to be simple enough for its components to be easily remembered.	All users will log in to the app with their phone number and a password chosen in the signup process. The password has no requirements for special characters or uppercase/lowercase characters.
How will the older adult be able to find their driver if they are getting transportation from someone they do not know?	The drivers are required to enter a photo of themselves, a vehicle description (make, model), and license plate number. This will prevent an older adult from being unable to find their transportation. This can also mitigate physical safety concerns due to the fact that an older adult can easily verify if a driver is who they state they are in the application.
What if no drivers are available at the older adult's time of desired transportation?	Contact information for alternate transportation resources will be displayed on the driver selection page of trip planning.
How will an older adult know that their transportation has arrived or that their	Notifications will be sent between drivers and passengers in transportation requests, request acceptance and denial, and a driver checking in to state that the older adult's ride has arrived.

transportation request has been accepted?	

4.4 Tasks

1. Construct a UI/UX design
2. Set up development environments
3. Create schema and construct the database. Add some test data to the database to later simulate drivers and passengers.
4. Implement an account creation page consisting of a form for users to enter and submit their sign-up information. Store each user's information into the database.
5. Implement a login page containing text box inputs to receive the user's username and password credentials. Verify the credentials and take the user to their Home page.
6. Implement a Home page for passengers consisting of Go Home, Plan a Trip, and Resource buttons that navigate to their respective pages
7. Implement a navigation pane for passengers.
    a. Implement a Settings button that takes the user to a Settings page with the options to edit personal information in a form or to manage their notification preferences in a form as well
    b. Implement a Help button that takes the user to a form that allows a user to email a description of problems they are having with the app to a developer account. Test out email functionality.
8. Implement a Home page for drivers consisting of Available Now, Schedule Availability, and Send Friend Request buttons that navigate to their respective pages.
9. Implement a navigation pane for drivers.
    a. Implement a Settings button that takes the user to a Settings page with the options to edit personal information in a form or to manage their notification preferences in a form as well
    b. Implement a Help button that takes the user to a form that allows a user to email a description of problems they are having with the app to a developer account. Test out email functionality.
10. Implement the ability for drivers to send friend requests.
    a. Implement text input boxes for a driver to search for a phone number and/or full name. Implement a scrollable list of clickable items; each item will be passengers a passenger that matches the name and/or phone number combination entered into the text boxes.
    b. When a list item is clicked, add the ability for a driver to confirm or deny that the selected passenger is who they would like to send a request to. Ensure that the request contains the driver's information. If confirm is selected, send a friend request to the passenger.
11. Test that the passengers receive the friend requests that are sent by drivers.
12. Implement the ability for passengers to confirm or deny friend requests.
    a. Implement buttons for a passenger to either confirm or deny a friend request sent. Send a notification to the driver indicating when the friend request is confirmed or denied.
    b. If the request is confirmed, the driver should be added to a list of trusted drivers stored in the database. Test that the confirmed requests are stored in the database.

13. Test that the drivers receive the friend request confirmation or denial from the passenger account.
14. Store the older adult's location in the application when the older adult wants to plan a trip. Use one of the geolocation packages in React Native to get the location, and store it in the application.
15. Test that the starting point geolocator works correctly.
16. Implement the time and date, driver, and destination selectors for trip planning on the passenger accounts.
    a. Implement a text box for a user to start entering their desired destination. Implement a scrollable list that filters nearby locations based on the text entered into the input box. When a user selects an item from the list, save it as the destination location. Add a "Next" button to take the user to the next page.
    b. Implement a form with text inputs for the driver to enter their desired time and date of travel. Add a "Next" button to take the user to the next page.
    c. Implement a checkbox for the passenger to select whether or not they would like to see only drivers in their friends list. Implement a scrollable list of selectable drivers that have availabilities overlapping with the passenger's desired time and date of travel.
    d. When a driver is selected, implement confirmation or denial of the desire to send a transportation request to said driver. If confirmed, send a transportation request containing the details from the inputs to the driver.
17. Test that transportation requests sent from the passenger account arrive at the driver account.
18. Implement the ability for a driver to respond to a transportation request.
    a. Implement Confirm and Deny buttons for the transportation request. Send a notification to the passenger indicating the driver's response to their request.
    b. Test that the response is sent from the driver account to the passenger account.
19. In the case of a transportation request confirmation, sync the time and date of transportation to both the driver and passenger's Google calendar.
20. In the case of a transportation request confirmation, sync the time and date of transportation to both the driver and passenger's Apple calendar.
21. Test that the transportation details are correctly syncing to the users' calendars.
22. Implement the ability for the driver to check in once they have arrived at the older adult's location. This will send the older adult a notification stating that their driver has arrived. Test to see that the check-in notifications have been sent to the passenger from the driver.
23. Implement the functionality of the Available Now button on the driver's home page.
    a. This should update the driver's availability in the database to now. Test that the drivers are visible in the "available drivers" list.
24. Implement the functionality of the Schedule Availability button on the driver's home page. This will take drivers to a page containing a form that lists days of the week and time slots that can be filled out to indicate availability. The dates and times will be saved to the database.
25. Test that the driver's availability is saved to the database.
26. Implement the functionality of the Resources button on the passenger's home page.
    a. Display the resources as a scrollable list of selectable items on a Resources page. When an item is selected, an area below the item will expand to display a description of the resource and an icon to call the resource.
    b. Implement the ability for a passenger to call the resources in the list.

27. Test the passenger's ability to call resources in the list.
28. Implement the ability for a passenger to rate a driver after transportation if they would like to do so.
    a. After transportation, display a popup asking the passenger if they would like to rate their driver. Clear "yes" and "no" buttons will be present. If "yes" is selected, display 5 stars for the passenger to fill in along with a submit button.
    b. Store the driver's ratings in the database. Test that their ratings are stored.
29. Prepare for and submit final code and presentations.

4.5 Schedule

	11/8-11/18	11/9-12/10
	Final Proposal; Preparation for Development	
Emily	UI/UX Design Draft	Environment Setup, Database Construction
Patrick K	UI/UX Design Draft	Environment Setup
Patrick P	UI/UX Design Draft	Environment Setup
Alan	UI/UX Design Draft	Environment Setup
Sailesh	UI/UX Design Draft	Environment Setup
	Sprint 1: 1.17-1/23	
Emily	Task 4: account creation page	
Patrick K	Task 5: login page	
Patrick P	Task 6, 8: Home page for passengers and drivers	
Alan	Task 7a, 9b: Navigation pane - settings	
Sailesh	Task 7a, 9a: Navigation pane - settings	
	Sprint 2: 1/24-1/30	
Emily	Task 7b, 9b: Navigation pane - Help	
Patrick K	Task 10a: Friend requests - implement searchable/filterable list of passengers	
Patrick P	Task 10a: Friend requests - implement searchable/filterable list of passengers	
Alan	Task 7b, 9b: Navigation pane - Help	
Sailesh	Task 7b, 9b: Navigation pane - Help	
	Sprint 3: 1/31-2/6	
Emily	Task 10b, 11: driver sends friend request, test	

Patrick K	Task 10b, 11: driver sends friend request, test
Patrick P	Task 14, 15: React Native geolocator, test
Alan	Task 14, 15: React Native geolocator, test
Sailesh	Task 14, 15: React Native geolocator, test
	Sprint 4: 2/7-2/13
Emily	Task 16a: Trip planning - destination finder/selector, test
Patrick K	Task 16a: Trip planning - destination finder/selector, test
Patrick P	Task 16a: Trip planning - destination finder/selector, test
Alan	Task 12a, 13: Passengers confirm/deny requests, test
Sailesh	Task 12a, 13: Passengers confirm/deny requests, test
	Sprint 5: 2/14-2/20
Emily	Task 16b: Trip planning - date and time selector
Patrick K	Task 16c: Trip planning - driver selector
Patrick P	Task 16c: Trip planning - driver selector
Alan	Task 12b: Add confirmed friend request to trusted drivers in database
Sailesh	Task 12b: Add confirmed friend request to trusted drivers in database
	Sprint 6: 2/21-2/27
Emily	Task 16d, 17: Send trip request to driver, test
Patrick K	Task 16d, 17: Send trip request to driver, test
Patrick P	Task 22: Available Now, test
Alan	Task 22: Available Now, test
Sailesh	Task 22: Available Now, test
	Sprint 7: 2/28-3/6
Emily	Task 18: Driver response to trip request, test
Patrick K	Task 18: Driver response to trip request, test
Patrick P	Task 23, 24: Schedule Availability, test
Alan	Task 23, 24: Schedule Availability, test
Sailesh	Task 23, 24: Schedule Availability, test
	Sprint 8: 3/7-3/13
Emily	Task 20, 21: sync confirmed request to Apple calendar, test
Patrick K	Task 20, 21: sync confirmed request to Apple calendar, test
Patrick P	Task 19, 21: sync confirmed request to Google calendar, test

Alan	Task 19, 21: sync confirmed request to Google calendar, test
Sailesh	Task 19, 21: sync confirmed request to Google calendar, test
	Sprint 9: 3/14-3/20
Emily	Task 20, 21: sync confirmed request to Apple calendar, test
Patrick K	Task 20, 21: sync confirmed request to Apple calendar, test
Patrick P	Task 19, 21: sync confirmed request to Google calendar, test
Alan	Task 19, 21: sync confirmed request to Google calendar, test
Sailesh	Task 19, 21: sync confirmed request to Google calendar, test
	Spring Break: 3/21-3/27
Emily	-
Patrick K	-
Patrick P	-
Alan	-
Sailesh	-
	Sprint 10: 3/28-4/3
Emily	Task 22: driver check-in, test
Patrick K	Task 22: driver check-in, test
Patrick P	Task 26: Resources - menu and descriptions
Alan	Task 26: Resources - menu and descriptions
Sailesh	Task 26: Resources - menu and descriptions
	Sprint 11: 4/4-4/10
Emily	Task 28: rating system
Patrick K	Task 28: rating system
Patrick P	Task 26, 27: Calling resources from the app
Alan	Task 26, 27: Calling resources from the app
Sailesh	Task 26, 27: Calling resources from the app
	Sprint 12: 4/11-4/17
Emily	Search for bugs and clean up code
Patrick K	Search for bugs and clean up code
Patrick P	Search for bugs and clean up code
Alan	Search for bugs and clean up code
Sailesh	Search for bugs and clean up code

	Final Submissions and Presentations: 4/18-4/24
Emily	Final submissions and presentations
Patrick K	Final submissions and presentations
Patrick P	Final submissions and presentations
Alan	Final submissions and presentations
Sailesh	Final submissions and presentations

4.6 Deliverables

- User interface drafts
- Database design and schema
- Text file of the test data entered into the database at the beginning of the project
- React Native and Python code used in the account creation process
- React Native and Python code used in the login process
- React Native code for the passenger and driver account Home pages
- React Native and Python code for the page functionality of Settings, Notification Management, and Help components of the navigation pane
- Documentation of the results of testing the functionality of emailing a developer account using the Help button in the navigation pane of both kinds of accounts
- React Native and Python code used in driver accounts to send friend requests to passenger accounts
- Documentation of the results of testing the ability of the driver accounts sending friend requests to passenger accounts
- React Native and Python code used in the passenger account's ability to view a friend request notification and to respond to a friend request
- Documentation of results of testing the ability of the passenger accounts to respond to friend requests from the driver accounts
- React Native code used in geolocation
- Documentation of the results of testing the geolocator
- React Native and Python code used in the passenger selecting their time and date, driver, and destination, as well as the submission of the transportation request to a driver.
- Documentation of the results of testing that the transportation request was successfully sent to the driver from the passenger.
- React Native and Python code used in the driver's response to the transportation request.
- Documentation of the results of testing that the response to the transportation request was successfully sent to the passenger from the driver.
- React Native and Python code used in syncing details of the confirmed transportation requests to both the driver's and passenger's Google or Apple calendars
- Documentation of the results of testing that the confirmed transportation requests are synced to both the driver's and passenger's Google or Apple calendars.
- React Native and Python code used in the drivers' ability to set their availabilities to now and to schedule their availabilities for the future
- Documentation of the results of testing that the drivers' availability changes are reflected both in the database and in the older adults' list of available drivers

- React Native and Python code used in the Resources page
- Documentation of the results of testing the passengers' ability to contact resources using the button in the Resources page.
- React Native and Python code used in the system for passengers to rate their drivers.
- Documentation of the results of testing that passengers' ratings of drivers are reflected in the database.
- A zip file of all final code submissions
- A zip file of all test documentation
- A zip file of all drafts and mockups
- Final presentation submissions

5.0 Key Personnel

Emily Lea - Emily Lea is a senior Computer Science major in the Computer Science and Computer Engineering Department at the University of Arkansas. She has completed the following courses that may be relevant to this project: Software Engineering, Database Management Systems, Computer Networks, and Operating Systems. She has six months of experience working as a Junior Consultant in software development at Rural Sourcing, Inc. She will be responsible for the initial database design and construction and will help out on backend or frontend after.

Patrick Page – Patrick Page is a Senior Computer Science/Computer Engineering major at the University of Arkansas. He has completed courses relevant to this project such as Programming Paradigms, Software Engineering, Database Management Systems, Information Retrieval, Mobile Programming, Computer Networks, and Operating Systems. He will be responsible for front-end mobile app development, UI/UX design, and connecting the front-end to the back-end when the time comes.

Sailesh Sirigineedi - Sailesh Sirigineedi is a senior Computer Science major in the Computer Science and Computer Engineering Department at the University of Arkansas. He has completed Programming Paradigms, Software Engineering, Database Management Systems, and Operating Systems. He also has experience from summer internships of both front end and back end as well as some database knowledge.

Alan Torres – Torres is a senior Computer Science major in the Computer Science and Computer Engineering Department at the University of Arkansas. He has completed project-relevant courses such as Programming Paradigms, Software Engineering, Database Management Systems, Computer Networks and Operating Systems. He has four months working with website UI/UX design through an internship from Spring 2021. He also has a few months working with database design/implementation and REST APIs through another internship from Summer 2021. He will be responsible for helping out with any database design aspects, UI/UX design for the mobile app, and working on any back-end and front-end implementation when needed.

Patrick Karangwa – Patrick Karangwa is a senior majoring in Computer Science at the University of Arkansas. He has taken classes like Programming Paradigms, Software Engineering, Operating Systems, and is currently enrolled in Database Management Systems Management, Algorithms, and Data Mining, which will all serve as background knowledge for this project. He also has background knowledge in static web application development through various different personal projects he created. For this project, he will primarily be working on the front-end side of the development process, although he will also jump back and forth to the back-end as well.

Dr. Suman Mitra – Dr Suman Mitra, our project sponsor, is an Assistant Professor in the Department of Civil Engineering at the University of Arkansas. His research focuses are the travel behaviors of disadvantaged populations, travel demand modeling, and sustainable transportation, shared mobility services and autonomous vehicles, transportation and health, and land use-transportation interaction.

Dr. Matthew Patitz - Dr Matthew Patitz, our project advisor, is an Associate Professor in the Computer Science and Computer Engineering Department at the University of Arkansas. His research focuses on DNA computing and algorithmic self-assembly. He received a CAREER award in 2016 and has published over 60 peer-reviewed journal and conference papers.

6.0 Facilities and Equipment

For the project we will be using our individual computers, Git, GitHub, React Native, Python Django, and PostgreSQL in the construction of the application and toolkit. We anticipate using packages from these technologies as well. To test the application, we will need at least one Android mobile phone and at least one iPhone.

7.0 References

[1] "Demographic Changes and Aging Population, https://www.ruralhealthinfo.org/toolkits/aging/1/demographics#:~:text=Between%202020%20and%202030%20alone,65%20years%20old%20and%20over.

[2] "Which US States Have the Oldest Populations?", https://www.prb.org/resources/which-us-states-are-the-oldest/

[example] Authors, "Article in Title Case," Conference or Journal, Publisher, Year

[3] "8 Ridesharing Services For Seniors", DailyCaring,

https://dailycaring.com/8-ridesharing-services-for-seniors/

[4] GoGoGrandparent,

https://gogograndparent.com

[5] O'Brien, Sara, "Lyft Focusses On Seniors With New Option To Book Rides By Phone Call", CNN Business, 2021,

https://www.cnn.com/2021/02/24/tech/lyft-call-a-ride/index.html