Demo Script

Flow

Intro
User story (problem statement)
App overview
Target audience
Main features (Use case models, Sequence diagrams, System Architecture)
Design patterns/principles
Live demo
Testing
SE Practises

Intro & Overview

Good afternoon TA and fellow classmates. My name is …, and as part of team BlaBlaBus, together with my groupmates …, we will be sharing with you our brand-new creation.

Next Slide

Now, imagine this. You're on your favourite messaging app – Telegram, Whatsapp, WeChat, whatever it is – and in the group chat with your friends, they decide that they want to hang out this Sunday. So everyone agrees on a location, say Suntec, and decides what time to meet, maybe 3 p.m.

Next Slide

But you, and everyone else, knows that this one friend is going to be late! He's always at least half an hour late, wasting everyone's time by making them wait for him. Wouldn't it be nice to have some way of knowing whether he is going to be late this time, and ensuring that he leaves on time?

Next Slide

Introducing OTWLah, our transformative real-time Singaporean transportation app designed to connect users, ensuring that they are never late again. With our top-of-the-line algorithms to minimize outdoor waiting time, coupled with our destination weather prediction system, users can not only ensure that they will always receive the best experience while travelling to their destination, but also have the peace of mind that whoever they are meeting won't waste their time.

Our target audience is teenagers and young adults who often make group plans to meet somewhere. However, these are not the only users of our app. Anyone, young or old, who

needs to make plans with others can use our app to make sure they know exactly when to leave.

Main Features

Next Slide

Now, I'll be sharing about the main features of our app. Firstly, any user can input their destination and preferred arrival time into the app. Upon submission, the app displays the time till the user needs to depart from their current location and the optimal route they should take to get there. This feature is known as solo travel mode, allowing users to use our app even if they don't have a party to join.

The next feature allows the user to check the weather at their selected destination, ensuring that they are well-equipped with the knowledge of whether their plans need to be changed due to wet weather.

Next, we have our party mode feature, allowing users in solo travel mode to turn their current travel instance into a party. From here, the app provides a QR code or a party pin number for other users to join the party. An edit party option is also available for the leader if the group changes their mind about the destination and/or arrival time.

Lastly, our app has a party history feature. Users can choose to use our app as a planner of sorts, allowing them to join many parties concurrently, and accessing them again through the party history page.

Next Slide (Show use case diagram)

Here is an overview of our app with all the use cases. The general flow is every user must register for an account, allowing them to log in using their email or phone number with a password or OTP. Once logged in, the user can use all our app's features as described above.

Next Slide (Show sequence diagrams for the features)

Delving deeper into the features described above here is the sequence diagram for the solo travel mode. Upon login, the user will be presented with a map view and an autocomplete search bar to find their destination. After selecting the destination, the user will then input their preferred arrival time. Our app then calculates the estimated departure time based on real-time traffic and public transportation data to provide the optimal route for the user.

Next Slide (Show party creation sequence)

At this point, the user has two options: continue their journey in solo travel mode until they reach their destination, or select the create party button to create a party using their travel mode

instance. If the user creates a party, they will be shown both the party QR code and PIN to allow their friends to join the party.

From this point, any user can join the party and the journey will proceed until everyone has reached the destination. However, if the edit party button is clicked, the set destination and arrival time sequence will be used to update the new party details. The route and ETA for each of the party members will then be recalculated and displayed accordingly.

Next Slide (Show party mode participant sequence)

Our next function allows users to join a party through different means. As mentioned earlier, the user can either scan a QR code or enter the party PIN to join a party. As long as the QR/PIN is valid, the user will be connected to the party.

Users can also concurrently join many parties at once, using the party history function to join the correct party at the correct time. This way, parties can be made in advance, without the need for the user to stay in the party when it is not necessary to. This feature comes in handy when you're juggling various social events or travel plans. Upon selection of the party in the list of parties, the user will then be connected to the corresponding party.

Next Slide

Now, let's talk about the weather feature. After selecting their destination, users can check the weather at that location whether in solo travel mode or party mode. The app provides real-time weather data, including temperature, humidity, and weather conditions. This helps users make informed decisions about their plans and make any necessary adjustments.

Next Slide (Show navigate to destination sequence cause fetch weather is part of it)

The sequence diagram for the weather feature is straightforward. The app sends a request to a weather service API, and the API responds with the forecasted weather data for the selected destination and time. Our app then displays this information to the user.

Next Slide (Show class diagram)

Here is our class diagram. Each class is defined by either a boundary, controller, or entity, performing its own individually unique actions. As you can see, once the account controller validates the user information, it calls the main activity to allow the user to access all our app's features. Our app makes use of 3 main controllers, an account controller for user validation, a party mode controller for party management, and a route controller for route calculation.

Next Slide (Show system architecture)

Moving on, this is the client-server architecture for our app. On the client side of our application, the main bulk of the front-end is formed by the views backed by the view controllers. The view controllers are responsible for the interaction between the client and the server side by sending HTTP requests to the various APIs as well as accessing our entity objects created on a mySQL server.

Design Patterns & Principles

Next Slide

Now let's look at the design principles that our team used during the development process. We decided to use the boundary-control-entity architecture in order to create a clear segregation between the different classes. This helps us to achieve high cohesion – as each individual class serves a well-defined purpose.

Next Slide

Our Party Mode Controller is also implemented as a Singleton. This is to prevent multiple unwanted instances of the controller as each client must have only one controller tied to it. It also allows multiple boundary classes to make use of the same controller instance whenever necessary.

Next Slide

In our notifications and location update methods, we make use of the observer pattern as our controllers will not know when there will be a new notification or location update. Hence, having an observer pattern to update whenever new data arrives allows for the desired loose coupling between objects.

This will also be useful for our future implementation of user settings to allow users to unsubscribe from notifications whenever they please.

Next Slide

We also obeyed the principle of least knowledge by separating the client and server code. Our client does not need to know the implementation of our server code and merely interacts with the server through HTTP requests to obtain the data that it needs. This level of abstraction improves the extendability of our code, allowing for new features to be added to either the client or server without affecting the other.

Next Slide

The Don’t Repeat Yourself (DRY) principle was also adhered to as many features of our solo travel mode and party mode overlap, such as route calculation, directions, and ETA. Therefore,

we included commonly used blocks of code into functions and classes for our controllers to make use of.

Live Demo

(Enter random info into log in page)

Now we will proceed with the live demo. Assuming that we are a new user, entering any account info here will produce an error as shown. That's why we must now go and create an account.

(Show register)

We will now proceed to enter the details of our account.

Username : xxx
Email: abc123 (invalid email)
Phone: 12345678 (invalid sg phone)
Password: abc (weak password)

As you can see, the email, phone, and password fields all show errors as they are not valid entries. Now if we input valid entries… an OTP will be sent to my email.

Username : xxx
Email: abc@123.com (change to one of our emails for otp)
Phone: 82314523 (make sure it starts with 8 or 9 and has 8 numbers)
Password: P@ssw0rd

(OTP is sent to the email)

Once we enter the OTP we received from the email into the app, our account will be registered and we will be automatically logged in.

So let's take the first example. You and your friends want to meet at Suntec on Sunday at 3 pm. So you log on to OTWLah and you're immediately able to search for your destination.

(Show destination search)

Our autocomplete suggestions function allows for quick and hassle-free destination input. As you start typing, the app intelligently predicts and displays potential locations, streamlining the process of selecting Suntec or any other destination. This ensures precision in specifying your meetup point, making the overall planning experience efficient and user-friendly.

After selecting one of the suggestions, OTWLah shows some details about the location you selected, allowing you to be sure that you have selected the correct destination.

(Show arrival time selection, select next Sunday 3pm)

After confirming your destination, OTWLah now prompts you to specify your preferred time of arrival. In our case, it'll be next Sunday at 3 pm.

(Click submit and slide the sliding panel up to show all info)

As you can see, OTWLah will provide you with all the necessary info for you to reach your destination on time, telling you how long till you need to leave your location, as well as the directions to get to your destination.

And if you ever get lost on the map, our current location button can be used to move the camera to where you currently are!

(Press current location button)

Now, since we need to invite our friends too, we need to create a party. Simply press the Create Party button…

(Press create party button)

And a QR code and PIN number will be generated, allowing your friends to join. If you bring your attention to our other phone, our friend here can now use the scan QR code feature to scan the provided QR code to join our party.

(Scan the QR code using other phone)

Now, in just a second, both UIs will be updated to show that someone has joined the party!

If you want to know where your friend is on the map, you can simply click on their user icon down below to find out where they are.

(On both phones, press on the other user's icon)

Moving on to our check weather function, clicking on the weather button allows you to see what the weather conditions are like at your destination

(Main phone press the weather button)

*Depending on the weather…
If (raining/cloudy){
"Looking at the forecasted weather, we may need to make some changes to our plans"
} else { "Looking at the forecasted weather, it looks like a perfect day out with friends!" }

*If it's raining/cloudy, segue to edit party since we need to “change plans”, if not, “Now say one of your friends hates Suntec, and wants to change the meeting location to VivoCity.”

As the party leader, you are given the option to edit the party details

(Show edit party)

We will be given the option to choose the same location, in case we only wish to change the arrival time, or, as demonstrated, we can change the location to VivoCity.

(Change location to VivoCity)

Again, the time will be set to our current arrival time in case no changes are required, which is what we will do. Of course, we can also change the arrival time if necessary.

(Don’t change the arrival time and submit the edit)

Now, we can see that OTWLah has provided a new route for each party member, as well as updated the ETA for everyone.

Say, for instance, we are already late for an event. We will simulate this by joining a party that has been set earlier.

(Both phones close party and join party number xxxx using party PIN)

Here, we can instead use the Party PIN to join a party if we please.

(Once both phones joins party)

As you can see, OTWLah lets you know whenever you have missed your departure time, with a small bell that appears under the corresponding user.

If we go ahead and click on the bell, a notification will be sent to that user to let them know that they should leave.

(Main phone press the bell on other user and show notification on other phone)

And now, you may be wondering, “what if I want to join multiple parties at the same time?” Well, OTWLah solves that problem by providing a party history tab. If we make our way back to the main screen

(Main phone closes party)

Under the menu button at the bottom right, we can access our party history. So any party that we have previously joined will be shown in this list

(Click on party history)

As you can see, both the party we created and the party we joined are on the list. If we wish to leave a party, we can simply swipe left on the party we want to delete and press the delete button.

(Swipe left on the 2nd party – the one we just joined – and delete it)

And of course, if we want to rejoin a party, we can just click on the party that we want to rejoin!

(Click on the party we created at the start)

That is the main gist of our app OTWLah. Moving on, we will now talk about the Testing we went through to ensure our app works properly

Testing

Next Slide (Show testing)

Our method of testing included both the concepts of black box and white box testing together. As seen, each test case is formed by a series of steps which execute the selected path based on the inputs, just like in white box testing. However, we also implement black box testing as each step of the test, every input includes one valid input from each equivalence class. The steps that are invalid also only contain one invalid input with the rest as valid inputs.

As an example, our register test case shows the steps for a user to register, but steps 2, 3, 4, and 5 include one invalid entry that produces the expected error, telling the user that their entry is not valid.

Good SE Practises

Lastly, our group attempted to adopt good software engineering practices during the development of OTWLah. We tried to implement the scrum methodology by including 2-week sprint cycles, checking back at the end of each sprint to see what was left to be done. We created a simple product backlog consisting of the individual features we needed to implement in order to stay organised and understand what we still needed to finish. Each person would select an item or a few items from the product backlog to try and finish by the end of the sprint.

At times, we also tried pair programming, either online or in person when possible. It allowed us to see the code from different perspectives, foreseeing issues that may have arisen in the future

if it wasn’t for the 2nd opinion. Bugs were also solved relatively quickly by having two people concurrently figure out what was going wrong.

Conclusion

And with that comes the end of our product demonstration. We hope you’ve enjoyed what we have presented to you today and welcome you onboard to make OTWLah a success. Thank you all for your time.