Peewee Script

# Intro <Eugenia> (2min)

Slide 1: Good afternoon. We are group 2 and today, we will be presenting to you PEEWEE, an innovative web application crafted to revolutionise the daily commute for frequent drivers in Singapore.

Slide 2: Here’s an overview of what we will be covering for our presentation today.

Slide 3: Now, we will start off with an introduction.

Slide 4: It was revealed that out of a total of 150 hours spent on driving, a staggering 56 hours were consumed due to congestion. Additionally, the study indicated that out of a total expenditure of 1118 SGD, approximately 227 SGD was directly attributed to traffic congestion. During peak hours, the average travel time for a distance of 10 km stood at 23 minutes, in contrast to the 16-minute average recorded in 2022.

PEEWEE is tailored to cater to the needs of our target audience, frequent drivers, and stands as an exceptional technological solution. Our ultimate objective is to demonstrate how this pioneering application not only enhances the daily lives of drivers but also epitomises the ethos of harnessing technology to redefine the way we live, learn, and work in Singapore.

Through this presentation, our aim is to underscore how PEEWEE harnesses the LTA API to generate the links to the latest images from traffic cameras all around Singapore. LTA’s Traffic Images are updated every 20 seconds for all 90 traffic cameras islandwide.

The PEEWEE app uses this API in its core function, along with our own traffic AI, to generate traffic trends.

Slide 5:

The model was also trained on night images as the untrained YoloV8 model was able to detect cars in clear daytime images but not any of the night time images or the smaller cars in the background.

Slide 6: As you can see from the differences in our untrained and trained images, our PEEWEE traffic AI model is able to produce much more accurate results. Now ill pass my time to Chengyao.

# 

# UI Demo

## Authentication <Cheng Yao>

## We will be doing a live demonstration of our PEEWEE Web App. Feel free to try it out on mobile or web by scanning the QR code on the screen.

## Peewee has been deployed on our server, so it is available for public access. You can take a picture to save it for later

## // Visit home page Lets start by running through Peewee’s authentication layer

Upon landing on our Peewee home page, new users can choose to register for an account, while existing users can choose to login to their accounts.

// Visit /dashboard

Peewee has a user authentication system set up for all our pages, so if unauthorised users visit our secured pages, they will be bounced out to the login screen where they would need to login to authenticate.

// Bounced to login

If I am a first time user, I would need to register for an account.

// Choose Register

Click on register, and key in my details for my account

(Register with email and password)

(Pause for Guang to enter details)

I will enter my email, followed by my password and confirm my password

// Forget password

Now, let’s say I am an existing user, and I have forgotten my password.

I can click on forget password, where i enter my email for confirmation.

An OTP will be sent to my email, and upon receiving it, I will enter it to validate my credentials (OTP is default 510210)

Upon successful validation, I will then reset my password.

// Set to new password

// Login after registering

We will then proceed with logging in.

## Dashboard <Zi Qin>

After users login to their accounts, they will be automatically directed to the dashboard page where they can view a summary of the traffic conditions and utilise shortcuts to other pages.

This graph on the top left allows them to view current real-time car count, past traffic trends and the estimated current congestion level calculated based on the average car count from past trends. These car counts are generated from the object detection AI model that we introduced previously.

Users can also visualise the congestion levels in a heatmap and navigate to the map page to view congestion levels along specific roads.

The dashboard also provides an overview of the recent road incidents and their favourite routes.

## Incidents <Zi Qin>- [Navigate to Incidents page]

Next, the incident page allows users to view the list of incidents that were reported by other users or themselves. This platform allows users to exchange information about road incidents that can affect congestion levels.

Users can also report an incident they encountered. They can do so by entering the incident type and a short description of the incident. Users will then need to grant system’s access to their location to allow the system to detect the location. After filling all the sections, the submit button will then be displayed for submission. The system then displays the submission status. After an incident is reported, the incident page will be updated and the map would also display the location on the map.

Demo

Incident Type: Accident

Incident Description: Vehicle collision on the rightmost lane

Incident Location Access: Yes

## Map Overview <Guang>

The map page displays the traffic conditions and traffic incidents. Traffic Cameras are plotted on the map and upon clicking a camera, an info window pops up showing the camera details, such as the Camera Name, Current Vehicle Count, , Peakedness level, and a link to view the life camera feeds in the Road Conditions page. Traffic Incidents are also plotted and its InfoWindow will show the type, description and location of the incident.

A heatmap can also be enabled, showing the current peakedness levels of a general location based on the traffic camera’s data.

## Map Filters <Aaron>

//Introduction to filters

"Today, we're looking at our map's two main filter sections which are used to help users see through the lens of traffic cameras and heatmaps, and a system to navigate around incident sites.

//Interactive Demonstration (If possible, show rough example)

As we interact with the map, notice how selecting different filters affects the displayed data. For instance, enabling the 'cameras' filter instantly populates the map with camera icons. Clicking an icon pulls up a live video feed, offering invaluable insight into current traffic flow and weather conditions.

//Camera, heatmap filters simplified

First, let's explore the camera and heatmap filters. These tools are designed to enhance situational awareness and provide real-time visual data. With the camera filter, users can locate live feed cameras, enabling them to visually assess traffic conditions. Switching to the heatmap view, one can immediately identify areas of high congestion, represented by colors, starting with red indicating heavy traffic, and with green signaling clear roads.

//incident filters

Moving on to incident filters. This set of filters is crucial for trip planning and navigation. Users can filter by incidents, which are unexpected events like accidents or hazardous conditions. Roadworks, another category, reveals planned construction zones, while closures provide information on roads that are currently not in use.

//Layered Filtering

Our map allows layered filtering, which means users can apply multiple filters to tailor the information to their specific needs. For example, by selecting both 'incidents' and 'closures,',etc, a driver can anticipate areas to avoid, ensuring a smoother journey.

//Conclusion

With that, i would like to pass onto hamka who will elaborate more on road routes

//Past Examples

1. we have 2 sections of filters, one for cameras and heatmap and another one for any road blockages reported, categorised by incidents, roadworks and closures.
2. Lets take a look at how the filters presents the map page.

## Map Routes <Hamka>

//Bedok to NTU

For routes, we can key in any two locations in Singapore. For example we can key in bedok and NTU here on the map. As you can see, the results are localised to Singapore and no results from other countries will pop up.

//Search route

Once we click search route, we can see the route plotted on the map.

//Save route

We can then save these routes to favourites and we can see it show up once we view favourites.

//Jewel to NTU

We can also search for another route such as jewel to NTU, and the new route will be plotted instead.

//View Favourites, plot Bedok to NTU

We can then plot our favourite routes on the map as well just by clicking view on map.

//Unfavourite the route

We can go back and decide to unfavourite these routes if we need to.

## Road Conditions <Eugenia> (click on view Camera from maps page)

1. From the map page, hovering over one of the cameras and clicking on” view camera”,will redirect us to the road conditions page.
2. On the road conditions page,camera details, the traffic camera image and a graph of average cars at each camera throughout the day is rendered, this allows users to visualise the trends clearly.
3. Now, without redirecting from our map, our road conditions page will simply display 4 live active cameras and its details on the screen.
4. Additionally, we may navigate to the camera details by using the search function to find another camera name.
5. Finally, we have also made our web application responsive on mobile versions so that it will be more convenient for users on the go. Now we are done with our demonstration. Ill pass my time on to Ziqin.

Back to Slideshow

# SE Practices <Zi Qin> (2 mins)

Moving on, I will be sharing some design practices that we have utilised in the development of PEEWEE.

We used a layered architecture system that groups modules into presentation, app logic and persistent data layers.

Within each layer, there is also clear segregation of responsibilities among the modules with modules responsible solely for either authentication, incident report, favourite routes, map or traffic conditions.

This ensures that there are only dependencies on related modules across the layers, keeping the coupling loose.

The loose coupling promotes maintainability by allowing modifications to modules to be made with limited changes to other modules. For instance, changes in the presentation layer do not affect other layers while changes to modules in the app logic layer and persistent data layer maximally affect 3 other modules.

In addition, we grouped common UI elements into components that can be imported to multiple other modules.

This promotes code reuse as the component only needs to be defined once and can be used by any other modules.

Changes to the implementation of UI elements in a common component will also be reflected in all the modules that utilise that component. For instance, when we modified the map design in the map component, the changes are directly applied to Dashboard, Map and View Route pages that display the same map component, saving us time and effort in implementing changes to the map display.

Using component also promotes code readability as it reduces whole sequences of codes into an encapsulated module for rendering.

These design practices would also aid in extending the functionalities of the app with minimal changes to existing implementation. For example, to include the functionality of computing and displaying travel time along routes based on current conditions,

the TrafficController would need to be modified to provide app logic in computing travel time. With minimal dependencies between TrafficController and other modules, modifications to the TrafficController will only result in modifications to MapUI and TrafficConditionsUI, leaving other modules unchanged. Furthermore, to display the travel time on the map, a new UI component can be created and this component can be easily imported and incorporated to the existing Map component used in the Map page. Therefore, these design considerations provide ease in extending functionalities of the app.

# Traceability <Cheng Yao>

Slide 18:

We now talk about how our project came together, from the planning stage to testing and deployment

Slide 19:

In our product development cycle, there are a few stages.

First, we discuss the requirements of our software and determine our functional and nonfunctional requirements.

Then we start with the design of our software, highlighting use cases and doing up our UI mock up.

After the design has been set, we begin working on implementing our ideas and designs into our actual product.

When all the components are implemented and put together, we then test our system for any flaws or unexpected behaviour before deeming it ready for deployment.

Slide 20:

Let us dive into details using our traffic conditions use case as an example.

The Traffic Conditions page allows the user to search for information on the traffic conditions at specific locations.

The user will be redirected to the traffic condition page after selecting to view traffic conditions on the navigation bar.

Slide 21:

In the planning phase, functional and non-functional requirements for Traffic Conditions are identified.

For its functional requirements, Traffic Conditions:

* Must allow users to view all real-time images retrieved from LTA’s Traffic Images API,
* And users must also be able to search and view images from a user-specified camera.
* Lastly, the users should also be able to view hourly and daily traffic trends **all** camera locations.

Slide 22:

Moving to the non-functional requirements:

* For the sake of web performance and user experience, our system must load the traffic trends in < 5 seconds.
* And for our data quality, our system must also detect vehicles in the images with an accuracy of **at least** 70%

Slide 23:

After finalising our requirements, we begin with designing and the implementation of our use case. Applying the MVC architecture, we have our Traffic Model, View and Controller

In addition to our MVC architecture, we have also adopted the loose coupling principle. By implementing our Traffic AI as a backend API, we separate the implementation of the AI from our controller, achieving loose coupling to reduce dependencies. This allows our Traffic Conditions and AI to both be more modular, being open for future extension - which can allow us to apply the open-closed principle.

Slide 24:

Through MVC architecture, we can apply the observer pattern more easily, with Traffic Trends subscribing to changes from our AI and the LTA API. When there are changes or updates in our Traffic AI, our Traffic Conditions controller will alert our model to update, which in turns notifies our Traffic Trends views to update, updating the Traffic Images and Trends on the user’s interface. The ability of these principles to work together directly addresses the functional and nonfunctional requirements of the traffic trends use case.

Slide 25:

After implementing our system and use cases, we begin our testing phase.

Using the White Box testing, we can effectively reduce the perceived risk of the system malfunctioning by testing out control flows.

When users select or search for a traffic camera, our system checks for the validity of the camera, if the camera exists, they can proceed to view detailed traffic trends. However, if it does not exist they will be prompted to select a camera again.

# Conclusion <Guang>

With its user-friendly interface and comprehensive set of features, PEEWEE empowers users to make informed decisions about their routes, saving valuable time and resources that would have otherwise been lost to traffic congestion. The integration of live camera feeds, along with detailed traffic analysis, allows for a holistic understanding of the road conditions, enabling users to plan their journeys effectively.

However, to make PEEWEE more indispensable for our drivers, we devised 3 possible improvements.Firstly, a feedback system can be integrated to cater to user concerns more effectively, to provide a user centric design. Secondly, real time alerts can be added to inform users of new incidents along their usual routes. And lastly, partnerships with local authorities and businesses can aid in providing better coverage of current road incidents, improving our data credibility.

We have come to the end of our presentation. I would like to end off with a quote by Ralph Waldo Emerson, an American philosopher: “It’s not the destination, but the journey that matters”. Thank you.