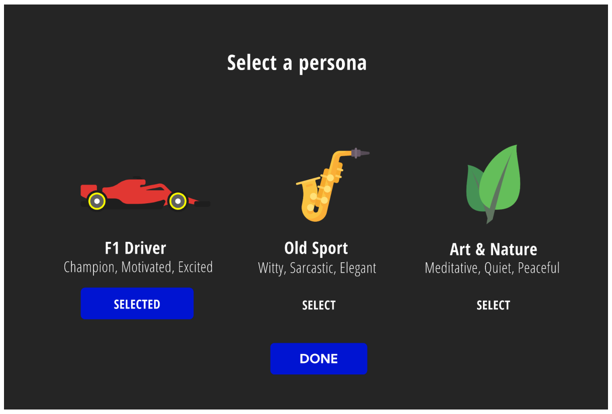
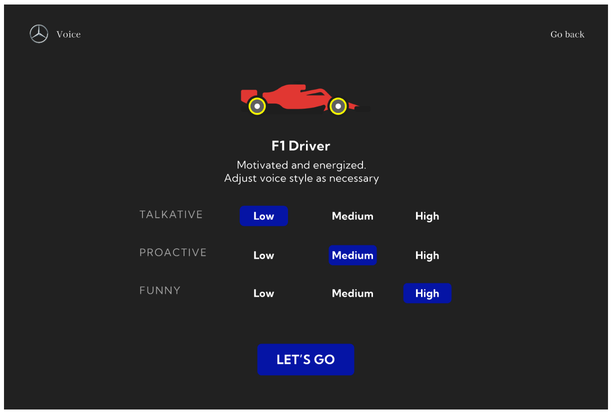
Stanford

The Mercedes Palette: Upgrading the infotainment system with GenAI personas

Product Overview

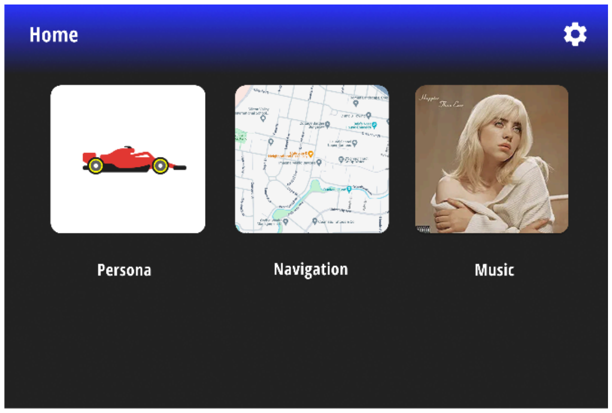
We propose a car experience where users can personalize rides by **choosing personas** for a unique and immersive Mercedes-Benz experience. **Personas** built with GenAI — conversation-focused LLMs and text-to-speech — accompany users on their journey, like a friend with character and personality.


Select a persona
F1 Driver
Champion, Motivated, Excited
SELECTED
Old Sport
Witty, Sarcastic, Elegant
SELECT
Art & Nature
Meditative, Quiet, Peaceful
SELECT
DONE



Motivated and energized.
Adjust voice style as necessary

TALKATIVE	[✓ Low]	Medium	High
PROACTIVE	Low	[✓ Medium]	High
FUNNY	Low	Medium	[✓ High]

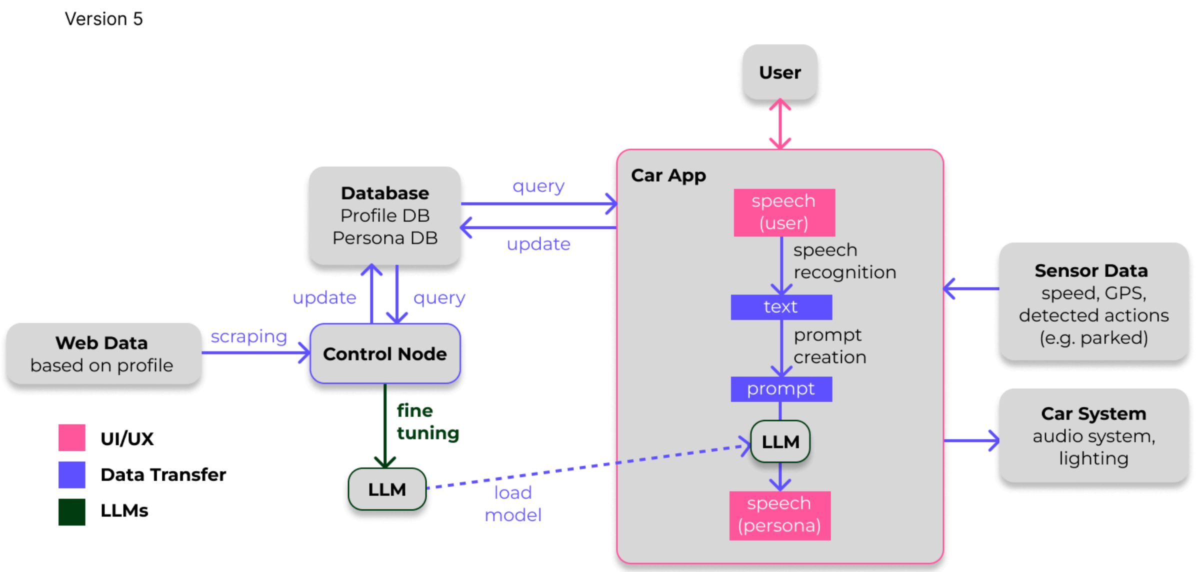


We have 3+ predefined personas, each with a specific **style**. Imagine choosing the *F1 Driver* persona. Race car sounds and dynamic lighting amplify your ride. Your persona speaks in the voice and personality of a renowned driver, not only guiding you with directions but also keeping you updated on F1-specific news, hyping you up during flawless turns and parking maneuvers, and injecting witty commentary with occasional references to racing jargon like pit stops, race tracks, and the Grand Prix.

Stanford

System Overview

We propose the following system with three compartments: a database, control node, and car app. See our System Architecture V5 document for more details on each part of the system.



Compartment	High-Level Functionality	Platform	Link to Code
Database	Remote data storage	Firebase	Firebase project
Control Node	Periodic news scraping LLM fine-tuning LLM storage LLM conversion	Python PyTorch	GitHub repo
Car App	Setup account Persona selection AI personality settings Speech recognition Prompt creation LLM query Speech output Link external accounts Parsing for requests Detecting events with sensor data Car system control	Android (Java)	GitHub repo

Problem and Differentiators.

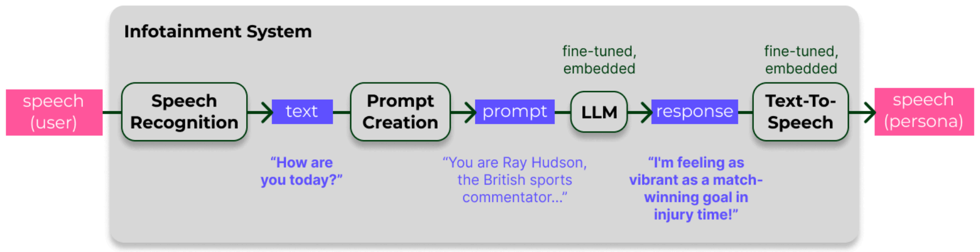
The MBUX system supports ChatGPT but is very SLOW. Currently, the workflow for MBUX takes in the user's voice, converts it to text, prompts the LLM on the cloud, sends it back into MBUX, then triggers the appropriate response. We will work with a smaller LLM to keep inference completely embedded. This will reduce latency because it doesn't require connection to a separate cloud compute source.

Existing AI assistants in cars do not focus on the "how" of personalization. Normally, personalization focuses on the "what" (favorite music, normal routines, etc.) rather than the "how" — *how is that information presented to the user?* Our personas, tailored to specific conversation styles and voices, make each ride fun and unique. Think of it as getting on a ride with different friends, each with a distinct personality and contributing to the ride in their own way.

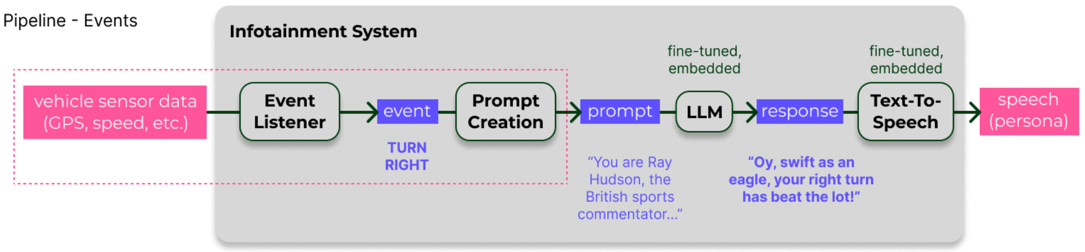
UX Flows

Here is the link to our Figma prototype of the user interface. Note that when selecting a persona, users are able to change certain personality attributes (verbosity, humor, etc.) manually if they choose to do so. Below are specific examples of the user interactions with the *F1 Driver* persona:

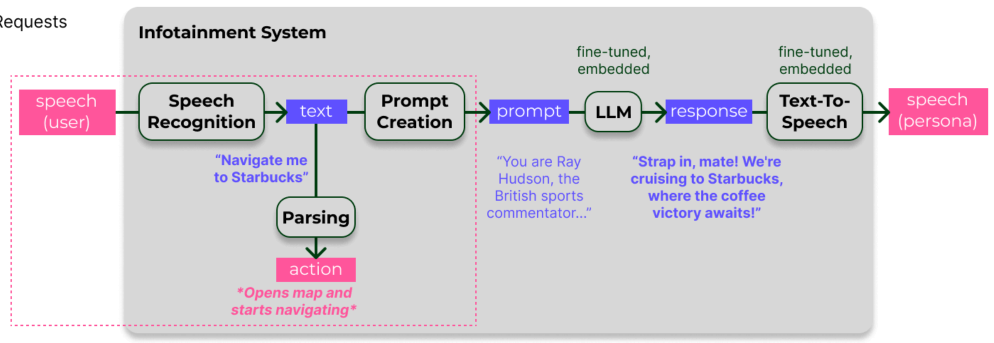
AI Pipeline



AI Pipeline - Events


Infotainment System
vehicle sensor data (GPS, speed, etc.)
Event Listener
event
TURN RIGHT
Prompt Creation
prompt
"You are Ray Hudson, the British sports commentator..."
fine-tuned, embedded
LLM
response
"Oy, swift as an eagle, your right turn has beat the lot!"
fine-tuned, embedded
Text-To-Speech
speech (persona)

AI Pipeline - Requests


Infotainment System
speech (user)
Speech Recognition
text
"Navigate me to Starbucks"
Parsing
action
Opens map and starts navigating
Prompt Creation
prompt
"You are Ray Hudson, the British sports commentator..."
fine-tuned, embedded
LLM
response
"Strap in, mate! We're cruising to Starbucks, where the coffee victory awaits!"
fine-tuned, embedded
Text-To-Speech
speech (persona)

Stanford

Functional Requirements

Compartment	High-Level Functionality	Description	Priority
Database	Remote data storage	Create and maintain two remote databases — **Profile DB** and **Persona DB** — that can be read and modified via the Firebase API.	1
Control Node	Periodic news scraping	Run a daily job that scrapes the latest domain news for each of our three personas, and updates the **Persona DB**.	4
	LLM fine-tuning	Fine-tune the LLMs to learn specific styles.	1
	LLM storage	Store the LLMs in Safetensor format.	1
	LLM conversion	Convert the LLMs into TorchScript for app integration.	1
Car App	Setup account	The user can sign up, log in, and log out of their account.	2
	Persona selection	The user can select from our predefined personas.	1
	AI personality settings	The user can adjust the AI's personality settings, including talkativeness, proactiveness, and humor.	1
	Speech recognition	The app should convert the user's speech to text.	1
	Prompt creation	The user's speech and passive events (e.g. turns, move seat, acceleration, parking) should be converted to a prompt, possibly augmented with the user's profile data.	1
	LLM query	The prompt, passed into our LLM, outputs a response.	1
	Speech output	Use a TTS model to turn the LLM's response into speech, with the voice tailored to the user's selected persona.	1
	Link external accounts	The user can link and unlink an external account, such as Spotify, Google Calendar, or GMail.	3
	Parsing for requests	The app can parse the user's speech, detecting specific functionality requests (e.g. navigation, music controls).	3
	Detecting events with sensor data	The app can detect events (e.g. turning, accelerating, passing objects) based on sensor data (e.g. GPS, camera).	4
	Car system control	The app can control navigation, music, and lighting.	4

Metrics for success

Objective Function

Testing fine-tuned conversation and text-to-speech models:

- **Minimize latency**, or response time from audio prompt to response (through model).
- Correctness of inference of the model (i.e. does what it's instructed).
- Minimal use of online queries to ChatGPT. Ideally, all requests handled by the embedded LLM.

Quality of response:

- Given a prompt to the LLM, how coherent is the response (i.e. does it make sense)?
- Given a prompt to the LLM, does the response embody traits of the persona (e.g. witty, poetic)?
- How well does the output of text-to-speech match our desired persona voice?

User adoption:

- How many users enable the AI assistant? What percentage of the time and which persona?

Goals

- Load Time Survey Results
- Acceptable Audio Latency
- Profile creation and updates are iterative, so it's hard to find a good baseline

Timeline

MVP completed by the end of Winter Quarter:

- **Profile DB** and **Persona DB** should be set up.
- **Control Node**: initial data collecting and fine-tuning pipelines should be set up.
- **LLM** should be tuned based on initial persona style (no incremental fine-tuning yet).
- **Car System**: Users should be able to manually toggle the AI system, change persona settings, as well as access typical car dashboard systems (music control, navigation).
- One persona (F1 Driver) fully developed.

Complete by Week 6 of Spring Quarter:

- Develop at least 2 more personas.
- **LLM fine-tuning**: Fine tune the embedded LLM to adapt to more specific styles.
- **Sensor Data**: Historical sensor data (e.g. location, speed, temperature) should be used to update the Profile DB.
- **Web Data**: Preemptively pull web data on news. Send difficult requests that can't be handled by the onboard LLM to ChatGPT. This should be a last resort.