Barrl, An Automated Bartending System

Erik Delanois
Computer Science and Engineering
University of California, San Diego
jedelanois@gmail.com

Gabrielle Evaristo
Computer Science and Engineering
University of California, San Diego
gevarist@ucsd.edu

Ted Schelble
Computer Science and Engineering
University of California, San Diego
eschelbl@ucsd.edu

Abstract

The COVID-19 pandemic in 2020 accelerated the need for contactless processes and experiences. The isolation of the pandemic created demand for safe but social interactions. To this end, we designed Barrl, an automated bartending system that allows patrons to order, pay, and pour their own drinks. We designed this system using a wine barrel body and a series of configured electronics in conjunction with a Raspberry Pi microcontroller. Experiments show that Barrl can produce 34 drinks an hour. From these results, we make the case for bar businesses to adopt a hybrid approach of human and automated bartenders to optimize revenue and patron wait times.

1 Introduction

For as long as there has been civilization, people have been coming together to share food and drinks. Many of today's modern conveniences have made it easier to create these social gatherings, but certain aspects of this experience have been negatively impacted by the contemporary way of socializing. Today, many people gather for events in public restaurants and bars. This has greatly improved the comfort of organizing engagements as the venue, food, and drink preparation can all be outsourced, requiring less effort from guests. However, as with all conveniences, there are drawbacks. Having to share venue space with dozens, and potentially even hundreds, of other patrons causes problems for both customers and establishment employees. With high volumes of people, crowds and long lines are inevitable. This can be an issue at restaurants, but is particularly evident at bars and venues, where customers need to stand in line to order food and drinks. Customers can be faced with multiple long waiting cycles for every round of beverages they order and incur additional wait times to pay and close tabs, all detracting from time they could be spending with their party. According to a study by Tails cocktails, the average Brit waits up to 12 hours a year for cocktails in bars in the United Kingdom [1]. Additionally, over a quarter have admitted to leaving for another bar due to excessive waiting. This amounts to easily lost revenue for businesses as customers are less likely to return. On the other hand, the need to juggle dozens of customers speaking simultaneously to order all at once can stress employees and limit the cash flow of the venue. Throw in a global pandemic, and these aforementioned issues are all exacerbated. Adding the need for social distancing, occupation limits on customers and staff, and limitations on business hours essentially destroy the current system. Customers are frustrated as their social lives are severely impacted, while businesses lose out on much needed revenue. Under these circumstances, there is little customer patronage at restaurants and bars. When there are customers, both customers and employees have to deal with the emotional

Preprint. Under review.

stress of potentially coming in contact with unknown people and becoming infected with a deadly virus.

Looking at the problems customers and venue employees face both within and outside of a global pandemic setting got our group thinking . . . there has to be a better way. Would it not be nice if people could enjoy their social events without having to take time consuming excursion trips to the bar and away from their party? Would it not be nice if venues did not have to throttle their revenue by having a limited number of bartenders assist only a handful of customers at a time? Would it not be nice if people could order a beverage in the middle of a pandemic and not worry that the person serving their drink could infect them with a deadly virus? These problems led our group to invent Barrl, a compact fully automated bartending system that enables users to browse, order, pay for, and pour drinks contactlessly through a safe and stylish web interface. Having one or multiple Barrl systems at a venue can significantly cut down on drink wait times, while providing business owners the ability to have a larger bandwidth in their drink serving abilities thereby increasing revenue. Another beneficial consequence of this automated product is that it does not require any human-to-human contact to operate, making it an ideal and even essential device for restaurants and bars during a pandemic.

Barrl has a number of competitors on the market. Products such as Bartesian [2], Somabar [3], and the Hamilton Beach COT2000 [4] are bartending systems that also attempt to ease the process of making cocktails, but all have severe shortcomings. Bartesian and Somabar are low grade consumer models that have a very low drink capacity and are intended for single person in home use. More industrial level competitors such as the Hamilton Beach COT2000 are expensive, only offer a limited number of drinks at a time, still require a bartender to operate them, and offer no customer payment portal. Barrl is the only sizable and scalable solution that provides owners the ability to create flexible and customized menus and customers the ability to order and pay for their own beverages. Our team was able to implement those functional features while also delivering a sleek web application and physical product.

2 Technical Material

2.1 Project Development

2.1.1 Development Roles

To go about building Barrl as a team of three, we elected to maintain equal roles as team members and in respect to the division of labor. In practice, this entailed all three of us working together to complete the physical fabrication aspects of our project as we all had varying levels of experience with both the hardware and software aspects of our project. In terms of software, we each took the lead on a different area of development. Erik built the embedded code that is responsible for data management and API services for interacting with ingredients, the menu, and pouring drinks. Ted built the owners portal, a secure front-end interface for connecting to the embedded code to manage the menu and ingredients. Gabrielle built the customer portal that allows customers to view the menu, edit their shopping cart, purchase an order of drinks, and pour them via embedded code functions. While each of us took primary responsibility over a software area, we all assisted each other outside of our primary focus (e.g. Erik helping Ted with a pesky Python bug, Ted systematically testing the customer front-end, and Gabrielle adding front-end notifications to all relevant areas of the web app).

As an additional note, when issues or discussions arose, we handled any disagreements by impromptu Zoom sessions and messaging via Slack to arrive at a consensus. In the rare cases that two team members did not agree on something, we took an informal vote and the majority ruled.

2.1.2 Hardware and Software Tools

The following table summarizes each hardware tool, their role in the project, and their price:

Hardware Tools	Role in Project	Cost
Raspberry Pi 4	Microcontroller that controls the pumps and manages the Flask front-end web server	$35
8 Channel Relay	Controls the Pi's signals and triggers the pumps	$9
12V Switching Power Supply	Powers the entire electrical system	$14
5V Regulator	Steps the voltage down to 5V for the Pi	$8
Power Distribution Board	Divides the electrical power into individual circuits for each pump	$17
Peristaltic Pumps	Transfers liquids from the bottles to the funnel	$70 for 6
High Voltage / Current Rated Diode	Prevents current from traveling backwards through the system when the system is turned off	$6 for 10
Food-Grade Tubing	Allows the liquid to flow from the bottles, up to the pump, and through to the funnel	$8
Kitchen Funnel	Allows the tubes to feed through to one opening	$9
Wine Barrel	Houses the dispensing unit, tubes, electronics, and bottle storage, and is the main physical component that owners and customers interact with	$75

Table 1: Main hardware tools

The two main software components include the Raspberry Pi firmware and the front-end web server. For the web server, we hosted Flask (a lightweight web development framework that functions like a REST API) on the Pi. Flask allowed us to mimic a web server so that users on the same WiFi network as Barrl can interact with the device. We used Python and JavaScript to control the logic and CSS and HTML to create an aesthetic interface for better user experience.

2.2 Process

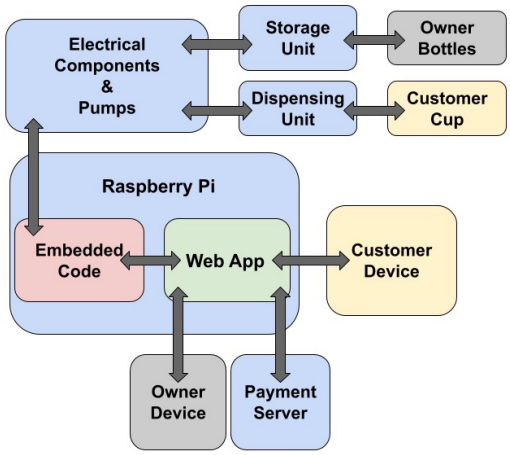

Figure 1: High-level system diagram

2.2.1 Wine Barrel

After researching and sourcing all of our materials, we spent three straight days working together in Gabrielle's garage to design, cut, and build our box, modify the wine barrel, and assemble the unit together. For owner ease, we cut through the backside of the barrel, creating a door that allows owners to easily replace or add liquids. Atop the barrel's body sits a customized wooden box that

houses the drink platform, liquid tubes, and electrical hardware. We partitioned the box into different sections to prevent the accidental leakage of liquid onto the electronics. We also built a shelf that sits within the barrel to keep the bottles closer to the dispensing unit, thereby shortening the needed tube lengths, and added wheels with a wooden base to make Barrl mobile. Our finalized physical device is shown in Figure 2.

Figure 2: Physical Barrl unit

2.2.2 Electrical Components

Our group put significant efforts in researching, locating, and ordering the proper electrical equipment. As a result, we were able to obtain reliable, relatively high quality electrical components for a fairly good price. The main electrical components are listed in Section 2.1.2. Once all of our components arrived, our group performed many exploratory, and at times risky, tests to determine how specific components operated as many of the units arrived without the proper schematics. Once we gained a thorough understanding of all components, we connected and soldered the circuit accordingly. When the circuit was completed and validated, components were fastened to a thin piece of plywood that served as our motherboard as shown in Figure 3. We then placed the board inside the backmost compartment of our top dispensing unit.

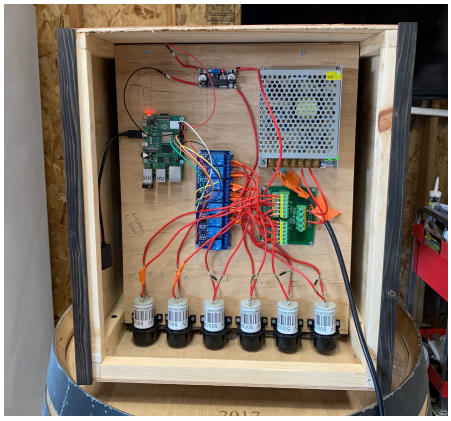

Figure 3: Electrical component board

2.2.3 Embedded Software

The embedded software is a central component that is leveraged by the front and back ends of the web server. It is responsible for managing stored ingredient and menu data, validating menus, and manipulating electrical hardware to pour drinks. We built the system with developer and user scalability in mind. We split the code into three main services: Ingredient, Menu, and Pouring. The Ingredient Service is responsible for implementing the functions that add, remove, and modify different ingredients to the database. The Menu Service leverages the Ingredient Service to allow owners of the device to create different drink menus based off of ingredients in the database, ensuring all drinks are composed of valid liquids. We implemented additional compatibility functions within the Menu Service to ensure that drinks displayed to the customer are compatible with the ingredients currently on tap. Lastly, the Pouring Service is responsible for taking a drink from the menu and sending proper instructions to the correct pumps to physically pour the drink.

Features of the embedded software include:

- Quick save utility functions to save any JSON compatible objects
- Ability to create, modify, and delete ingredients
- Utility functions to mark which ingredient is connected to what pump
- Ability to create, modify, and delete drinks from the menu
- Function that validates drink compatibility with the current ingredient database
- Function that validates drink compatibility with the current ingredients on tap
- Ability to pour drinks by their menu identification number
- Ability to pour all ingredients from a drink in parallel
- Mock Pi functions for development on non-Pi machines
- Pump cleaning functions that clear out the pumps for a specified amount of time

2.2.4 Owners Portal

For Barrl to be a useful product, there needed to be a portal for owners and bar staff to curate the customer experience and manage Barrl's inventory. Since the bar staff and management would be responsible for stocking the unit and owning the menu, we decided to build a one-stop shop for employees to reflect any changes to stock or ingredients and manage the venue's customizable menu. The resulting portal can only be accessed by a secure access code. Once authenticated, staff can add ingredients to their inventory, manage existing ingredients, add new drinks to the menu, and manage the existing menu. This portal serves to keep Barrl in a stable state as staff can use a simple front-end interface to leverage the embedded code for different data actions. For example, staff could swap in a new ingredient, like Tequila, which automatically enables drinks containing Tequila and other available ingredients to be displayed on the menu. This way, staff can make quick and easy changes on the fly that get reflected instantly for customers.

A consolidated list of the owners portal features include:

- Portal security so that it can only be accessed with a staff access code
- Mobile-friendly responsive design that staff can access via their phones
- Ability to add new ingredients to the existing inventory
- Ability to delete or modify depleted ingredients
- Ability to update Barrl's knowledge of which ingredient is connected to what pump
- Ability to add new drinks and their makeup to the menu
- Ability to modify the existing menu's drinks, names, descriptions, images, and prices

The interface for the owners portal is shown in Figure 4.

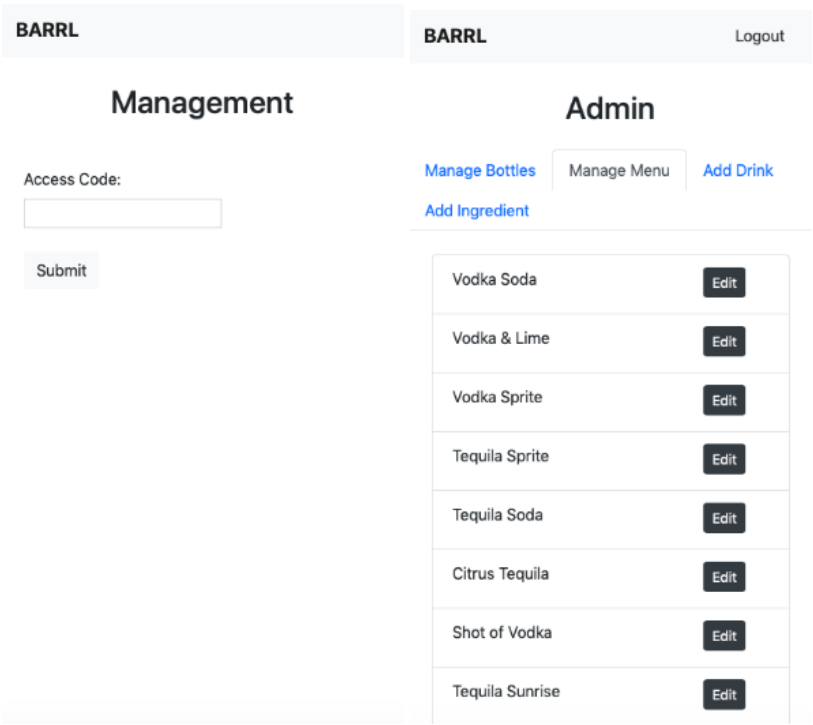

Figure 4: The left image is the admin login interface and the right image is the admin management interface

2.2.5 Customer Portal

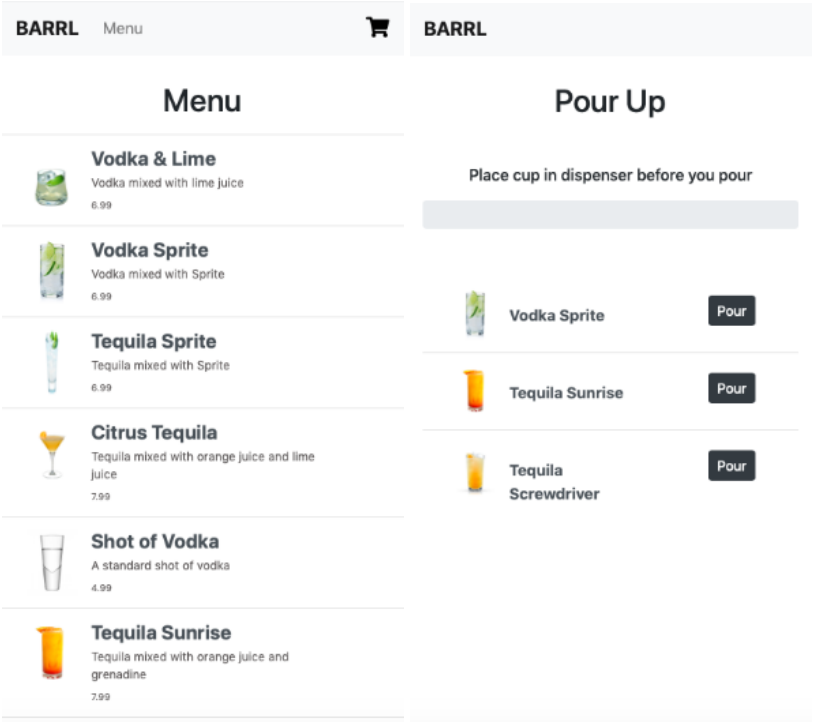

Figure 5: The left image is the menu interface and the right image is the pour portal interface

In addition to an owners portal, we implemented a portal that customers could interact with. To access the interface, customers scan the QR code on the Barrl device, which reroutes them to Barrl's online website. From there, users can select drinks, view their subtotal, pay, and have their drink poured. Our Flask application maintains customer sessions, such as the specific items in a customer's cart. Further, we designed an aesthetic, easy-to-use interface so that non-technical users can easily and effectively navigate through the portal via their mobile phone. The menu and pour portal interface are shown in Figure 5. As users interact with the front-end application, the API interacts with the Pi for pouring drinks to create an all-in-one experience via the mobile app and the physical device.

Features that customers can expect when interacting with the portal include the ability to:

- Browse through the menu
- Add drinks to their cart
- Edit the quantity of items in their cart
- Remove items from their cart
- Add a promotion discount code at checkout
- Pay for their drinks via Google Pay, Apple Pay, or a debit/credit card
- Pour their drinks by pressing a button on the pour portal page

2.2.6 System Integration and Testing

In the final week of the quarter, our team integrated the disparate sets of software that we had been working on. Since we had maintained an active GitHub repository (170 commits to date), our frequent integrations eased and simplified the process. However, even with frequent communication via Zoom and Slack, we had different mental models of how the necessary data would be managed and how ingredients would be added, updated, and deleted in relation to drinks and the menu at large. When Erik pushed his highly scalable code of managing ingredients and drinks, it did not match the logic that Ted used for management in the owners portal. We agreed that Erik's approach would yield a superior product and Ted refitted the owners portal to make use of the services that Erik developed.

Once the entire code base was functional together in local environments on our computers, we met in person to begin end-to-end testing. After pushing our latest code build to the Raspberry Pi within Barrl, we tested all of the administrative and customer functionality. Everything worked as it had on our local computers, a result of our repeated integrations on GitHub, local testing, and frequent communication. We poured several different drinks to evaluate time and tasting for consistency, and found that the slowness of our pumps actually yielded fairly consistent pours.

3 Milestones

As a team, we accomplished all that we set out to do. All milestones proposed in our Project Specification, with slight tweaks mentioned in our Milestone Report, were successfully implemented.

3.1 Accomplishments

3.1.1 Wine Barrel Assembly

Accomplishment	Person
Researched and ordered the barrel	All
Created a door for the barrel to allow bottle storage access	All
Built a shelf for the bottles to keep them closer to the dispensing unit	All
Constructed the wooden box for the dispensing unit, electronics, and tubes	All
Attached wheels to the bottom of the barrel for mobility	All

3.1.2 Configuration of Electrical Components

Accomplishment	Person
Researched and ordered electrical components	All
Initialized Raspberry Pi and confirmed basic code can be written to it	Erik, Ted
Set up ssh for remote access to the Pi through a local internet connection	Ted
Wired electrical components and ensured that pumps can successfully operate with electricity	All
Housed electronics in the Barrl unit	Gabrielle, Ted
Installed tubes and funnel	Gabrielle, Ted

3.1.3 Embedded Software

Accomplishment	Person
Initialized GitHub repository with base code	Ted
Created JSON files to store the ingredients and menu	All
Implemented the API's for the Pi to control the different pumps	Erik
Created code that allows the Pi to control specific amounts of each ingredient in a drink via the pumps to a certain degree of accuracy	Erik
Created code that receives drink orders from the customer portal	Erik

3.1.4 Owners Portal

Accomplishment	Person
Initialized Flask application	Ted
Created authentication for owners to allow access to non-customer related operations	Ted
Built the interface that allows owners to modify ingredients, drinks, prices, etc.	Ted
Wrote code that allows modification of ingredients, drinks, prices, etc.	Erik

3.1.5 Customer Portal

Accomplishment	Person
Built the interface that customers can see and interact with	Gabrielle
Wrote code that sends drink orders to the embedded code so that a drink order actually triggers the Pi's operations	Gabrielle
Integrated Stripe API to allow contactless drink payments	Gabrielle

3.1.6 System Integration and Testing

Accomplishment	Person
Integrated the embedded software and front end portals code	All
Performed end-to-end testing to ensure that all components functioned as intended	All

3.2 Challenges and Limitations

There were no significant challenges that our team was not able to overcome. With the COVID pandemic going on, our team was only able to meet in person with all team members for 3 days at the start of the quarter. Due to meticulous planning and the sourcing of all our materials, we were able to complete the physical aspects of our project early on. Our focused strategy to achieve this early allowed us to implement nice-to-have features that pushed Barrl from a proof of concept into a real product with market potential.

Even though we achieved our proposed goals, there were some challenges and limitations that we faced as team. These include:

- Different mental models of the system − While we successfully integrated our disparate areas of work by the end of the quarter, we would have saved a lot of time if we explicitly defined how each component integrated with one another.
- Needing significant physical space − Our team definitively lucked out by having Gabrielle's garage available as a workshop throughout the quarter. Without a dedicated space for fabrication that also had tools, electricity, and Wifi, we would have really struggled to make progress.
- Refrigeration − Currently, Barrl does not have an ice dispenser or any form of refrigeration for the ingredients. We rely on a separate ice bucket in which customers have to manually grab from before pouring their drinks. In hindsight, this is a limitation given how prevalent and necessary ice is for most cocktail recipes.
- Still slower than a bartender − We set out to build a device that would help reduce wait times. However, Barrl is still slower than the average human bartender; on average, it pours a standard cocktail in 105 seconds vs. 20 seconds for a human bartender. Though this result signifies that we would need 5 or more Barrl units to match one human bartender, it is worth noting that Barrl excels where a human bartender struggles − drinks with a higher number of ingredients. Since Barrl can pour from different bottles simultaneously, it is only limited by the ingredient with the highest volume. A human bartender can pour more quickly, but likely only one or two ingredients at a time. Therefore, when a drink has more than two ingredients, Barrl gets relatively quicker at pouring than a human does.
- Remote work and different team member time zones − As mentioned previously, this challenge was not significantly problematic for our progress, but it did prove to be inconvenient and difficult at times to schedule meetings that worked for the entire team. Working remotely necessitated additional care with communication as much can be lost in translation over Slack or Zoom.

4 Conclusion

Our team achieved the main goal of building a device that would allow customers to order, pay, and pour their own cocktails. Our Barrl device is ready for use as a real-life product in a bar or social event setting. Customers will be able to avoid crowds and lines at the bar, access Barrl's web app via scanning a QR code, browse the menu, pay without any human interaction or wait, and pour their drinks themselves. On the administrative side, owners can manage their ingredients through the Barrl door and electronically, via the owners portal, update their menu's items and prices and divert traffic from bartenders to Barrls. Ultimately, our core contribution is enhancing a customer's experience of purchasing a cocktail by putting the power to pay for and produce a drink in their hands. Customers would wait less, pay less (as robots do not need tips), and enjoy the novelty of having their drink poured at the press of a button. This utility is enabled by our physical Barrl unit in conjunction with the public code repository that our team published for public usage in similar products or projects [5]. To aid future developers of automated bartending systems, we offer our front-end web application for customers and managers, data management system for ingredients and drinks, and our embedded code that allows drinks to be poured with a single function call. For potential future iterations of the Barrl device, we identified several opportunities to remove limitations and increase Barrl's utility:

- Increase the number of dispensable liquids – This is the most intuitive improvement that can be made. Currently, the device is capable of dispensing six types of liquids as ingredients for cocktails. The number of spirits and mixers can be expanded, thereby increasing the number of beverages that can be created and poured.
- Reduce pour times – Presently, Barrl takes approximately a minute and 45 seconds to pour a standard cocktail, limited by the time it takes to pour the needed ingredient with the most volume. This is a reflection of our slow motors, which pump approximately 80 mL/minute. Faster motors could improve pouring times.
- Build more Barrls – For a full proof of concept, we could test how a "fleet" of Barrl units functioned together in practice with bar staff managing and restocking them. This could also involve new functionality like a more robust inventory management system or live sales reporting.
- Include a refrigeration and/or ice dispensing system – This is mentioned in Section 3.2.
- Build a cup dispenser – When a customer purchases a drink from Barrl, they must place their cup in the dispensing unit. Future versions could develop a cup dispensing system that would automatically and properly dispense and position a cup.
- Implement empty bottle sensing – With the proposed version, owners would have to manually check for any empty bottles. Future versions of Barrl can tackle this problem by utilizing sensors to detect when a bottle becomes empty.

5 References

[1] Morning Advertiser, Long queue times could drive customers away - https://www.morningadvertiser.co.uk/Article/2019/11/08/Long-queue-times-could-drive-customers-away-research-reveals

[2] Bartesian - https://bartesian.com/

[3] Somabar - https://www.somabar.com/

[4] Hamilton Beach COT2000 - https://budgetsupply.com/hamilton-beach-cot2000-cocktail-dispenser-machine.html

[5] Barrl GitHub Repository - https://github.com/tschelbs18/barrl

[6] Barrl Project Wiki - https://github.com/tschelbs18/barrl/wiki

[7] Barrl Google Drive - https://drive.google.com/drive/u/1/folders/1snSKSgJmyOvtk9PzmkVpTZoNSdSPx8OJ