6.1800 2026 Project Specification

1. Introduction

Bike share systems, like the Bluebikes you see around Cambridge and Boston, can make transportation less expensive and more enjoyable for citizens, and reduce pollution. In this project, your team will design part of an enhanced bike share system for the city of Newplace (pop. 1 million). Based on existing systems in other cities, Newplace expects approximately 50,000 citizens to become members of the bikeshare, and to have around 20 million completed rides per year. Newplace is currently accepting bids for this system. Your team works for Bikes4All, who would like to make a successful bid to design, build, and operate the system.

In addition to allowing riders to rent and return bikes, this enhanced system will include several features that you don't see in standard bike share systems:

- The ability for riders to reserve a bike for pick-up at the start of their ride, and to reserve a dock for drop-off at the end.
- Support for an experimental camera module on certain bikes.
- The ability for the system to remain working through certain types of communications and/or power failures.

The objective of the overall system is to meet the needs of the city and as many citizens as possible, while giving them an enjoyable and effective experience as well as respecting their privacy. Your team will be responsible for designing the subsystem that supports the experimental camera module. Another team — the Reservations Team — has designed the part of the system that gives riders the ability to reserve bikes. It's possible that the parts of the system that your team designs will need to interact with modules that the Reservations Team has designed. It is also possible that your team will need to propose changes to some of the modules that the Reservations Team has designed, particularly in the case where transferring a video conflicts with a bike reservation.

2. System Overview

2.1 Modules

The bikeshare system consists of bikes, stations containing a kiosk and set of bike docks, the central computing facility, and riders. In addition, there is a phone app that provides rider access to the system. The company that wins the contract — hopefully Bikes4All! — will handle the operation of technology on the bikes, stations including the kiosks and docks, the central computing facility, and the capabilities of the phone app.

2.1.1 Bikes

The bikes in this system communicate with the rider's phone running the app using either Bluetooth, cellular data or WiFi to provide common functions. Each bike is uniquely identified and has at least a limited capacity to report its current GPS location via a very low-capacity long-range radio system. Bikes are equipped with remotely controllable locks, and a small number of bikes have experimental camera modules. Your team is developing support for those camera modules.

2.1.2 Stations

Bikes are stored in docks at a station, and each station also has a kiosk at one end containing an interactive touch screen, a card slot for credit/debit and bike cards (see below), and a keyboard. The screen is used both for specifics about borrowing a particular bike and additional useful information such as the number of bikes and docks available at nearby stations.

Bikes are locked into a dock unless unlocked through confirmation of an account or payment. Once payment or an account is confirmed, the system will cause the dock to unlock the appropriate lock and can be adjusted for fit and ridden. To terminate a ride the bike must be inserted into a dock. Docks are designed so that the rider sits the bike snugly into the dock and the dock automatically locks and connects the bike to the USB and power plugs. At that point any exchanges between the bike and the system will need to occur — Bikes4All is designing those exchanges, and your team will need to deal with any that relate to transferring videos.

In case of power and/or communications failures, the stations should be prepared to continue some level of rider service. In support of that, each station will include battery back-up.

As a contribution to the system, Newplace will build enough stations for the bikes so that every station is within 1,000 feet of at least one other. Newplace is also prepared to support the electricity and communications infrastructure to connect the stations to a central location.

2.1.3 Riders

There are two broad categories of riders: members and non-members. Members have accounts and can be identified by either an app running on their phones, or a bike card provided by Bikes4All (only members can receive a bike card). Members may have a credit or debit card associated with their accounts, but the city would also like to allow members to pay with the equivalent of cash, by adding "money" to their bike cards. This can be done with cash payments at several public locations such as banks, post offices and public libraries. Thus, people can be members without having to have a phone or credit card.

Members can borrow a bike either with the phone app or a bike card at the kiosk. At present, they cannot use their credit/debit cards to directly access their membership. If they have only a

credit/debit card at a kiosk, they will be treated as non-members. Non-members — visitors, tourists, etc. — can only borrow a bike with a credit or debit card at a kiosk.

2.1.4 Centralized Computing Infrastructure

The centralized computing infrastructure keeps track of several types of information:

1. Basic management information. For example, user accounts, data about individual bikes and docks, etc.
2. Information needed to make long-range decisions. For example, decisions about the (re)placement of docking stations, the number of bike docks in those stations, the number of bikes in the system, etc.
3. Information required by the city both for monitoring the efficacy of the system and making decisions about road and traffic management.

Other teams at Bikes4All are responsible for maintaining these data sets. Your team will need to consider the storage of videos.

2.1.5 Communication between Riders and the System

For the bikeshare system to be useful, riders must be able to communicate with the system. Members can communicate with the system via the app and kiosks at the stations and do things such as borrow a bike. When on a ride, the only option for direct communication is through the app.

Non-members have one option for communication: the kiosks. At a kiosk, they can provide a credit/debit card and borrow a bike as well.

Your team is not responsible for designing the user interfaces for any of these modules, nor for handling the information flow required for checking out, dropping off, or reserving a bike. You will need to design any information flow related to the transfer of videos, though.

2.2 Existing features of the system

Overall, the system is meant to enable riders to easily share bikes effectively at a reasonable cost, while respecting their needs, expectations, and personal privacy.

The Reservations Team has already built out the basic functionality of the system that allows riders to rent and return bikes, as well as to reserve rides. This includes:

- Functionality for billing members for their rides, including when they ride past the end of the amount of time they've pre-paid for.
- Reservations for a user to pick up a bike, drop one off, or both.

3. Camera Module Enhancement

Your team is responsible for designing the subsystem that supports the experimental camera module on some bikes.

3.1 Experimental Camera Modules

Bikes4All is going to begin testing the inclusion of cameras on a small number of standard bikes and ebikes. The cameras are meant to allow riders to record personal videos along their ride and transfer those videos to their phones, and to record accidents. Accident videos will help Bikes4All understand riding skills and riding conditions, as well as estimate possible bike repairs needed.

Your system must send any accident videos to the central computing facility, though there is no time constraint on this requirement. Note that a user might return a bike with an accident video still stored on it, and another user may try to borrow that bike quickly thereafter (before your subsystem has sent the video to the computing facility). You will have to choose how to handle this situation, but the video should not be deleted from the video storage on the camera until the transfer to the computing facility is complete.

This scenario is possible with personal videos as well, but personal videos are not required to be sent to the central computing facility. You will need to decide how to handle their transfer, deletion, etc.

3.2 Failures

The city of Newplace has specified that the stations must remain at least partially available during a power or communications outage. Your team should decide to what extent video transfers should be supported during these failures.

3.3 Data Collection and Privacy

There is a lot of data that the system can collect that would be useful to both Bikes4All and the Newplace government. For example:

- How long are the rides? Where do they start and end? Where do they go?
- Is there a way to estimate how much of the riding is commuting vs. errands vs. riding for joy and entertainment? How frequently is each rider doing each type of ride?
- Is the system reaching underserved or lower-economic communities?

By gaining a better understanding of the bike usage of the roads, Newplace may be able to improve the safety and traffic management of roads, and Bikes4All may be able to adjust and prepare for further growth.

Many of these questions can be answered by using data collected from the GPS trackers and/or cameras on the bikes. However, GPS trackers and cameras often reveal personalized information. Your team should make recommendations for how video data should be stored given this context.

4. Detailed System Components

Here, we present more details about each component of the system, as well as some more detailed requirements needed to support the functionality described above.

4.1 The bikes

4.1.1 Identifiers

Each bike has a system-wide unique identifier (ID), which is just a unique number assigned to the bike. The ID will be written on the bike itself as well as encoded in a QR code that can be read in from the bike system app. In any place where the bike ID is transferred by the system, it will be stored as an integer (four bytes long).

4.1.2 Electronics Systems

Each bike has an on-bike electronics system that contains:

- **A GPS receiver**. For the purposes of this project, location information will have three components, the latitude (from the Equator) and longitude (from the Prime Meridian) in degrees as a floating point number, and the time at which it was taken.

- **A LoRa two-way radio system.** This radio system has a long range (1-2 miles) and a very low capacity (average 10Kbps, with a range of 0.3–25Kbps). The LoRa radio is supported by an internal battery and is off (not listening) for most of the time. If it wakes up to send and receive one message every 30 minutes, the battery will last for at least two years. If it wakes up more frequently the battery will last a shorter time. The LoRa system on the bike can also put itself into “listening” mode, so that it is listening continuously, but that takes yet more power; this mode should be used sparingly.

- **A remotely controllable lock.** The lock can be engaged or disengaged based on data it receives from the LoRa radio.

- **A simple “health” monitor for the bike**, to report any issues that may require maintenance or repair. This consists of two sensors that will measure tire pressure and breaks and notify the on-board processor of the status of the tires and brakes.

- **A simple processor** that can manage the storage of GPS information (up to one hour of GPS recordings from the receiver), receive and process the tire and brake “health” sensors, and operate the remotely controllable lock.

- **Additional storage**, so more information can be stored on the bike, such as route information, perhaps simple pointers for tourists, etc.

- **A connector used to “plug” into a dock.** This dock supports USB 3.1 (a limited capacity protocol, providing a maximum of 600MBps). When a rider puts a bike into a dock, the bike’s ID and health will be reported through this interface. In addition to any further GPS information not previously reported can be transmitted.

- **A small screen**, so that additional information can be provided to the rider during a ride.

- The bikes can transmit data wirelessly via **Bluetooth, cellular data, or wifi**.

Since the bikes have multiple wireless communications options, it’s important to understand the differences between them:

- **Bluetooth** is used for short-range communication (<= 30ft). It does not transmit through metal and many other building materials but has higher capacity than LoRa (packets can be up to 64 kilobytes long and 2.1 megabits per second). Because of the short range, Bluetooth data can only be sent from the bike directly to a user’s phone. There is no dollar cost for using Bluetooth and it can only be used to communicate directly with the phone.

- **Cellular data** can be used for longer-range communications; cellular devices can be up to 1-3 miles from a tower, although this is line of sight, so in urban areas cellular companies place “towers” much closer together, and there still are dead spaces. Bikes that use cellular communications will communicate directly through the cellular network; they cannot peer with a phone and communicate through the phone. Data can be transmitted at up to 80Mbps, and costs $50/month per bike. This service provides only direct communication with the central computing facility.

- **WiFi (802.11ax)** will achieve between 20Mbps and 200Mpbs (depending on location). The ISP has promised that they will provide enough hotspots that for all intents and purposes there are no dead spots, only lesser capacity areas. As with the cellular service, the devices cannot peer with phones directly using this Wifi. The local ISP has equipped the city with hotspots and contracts for this service are $75/month per bike. This communication can only support communication directly between the bikes and central service.

4.1.3 Cameras

Bikes4All is going to begin testing the inclusion of cameras on bikes. The cameras have two purposes:

1. To allow riders to record things along their rides and transfer them to their cell phones. After a bike with a camera is returned to a dock (within some limited time period to allow for extraction of a final recording), these videos should be cleared from the bike.

2. To record accidents. These videos should be sent to the central computing facility for analysis. Depending on the protocol for transfer, these accident recordings may remain in the camera for longer. You will need to decide whether a rider can access such accident recordings of their own rides.

Because the cameras are a new and untested feature of shared bikes, Bikes4All has decided that they will provide these on only 10% of the bikes. The cameras that Bikes4All are considering are medium quality, but highly ruggedized to withstand frequent use. Each camera will have two lenses, pointing forward and toward the rider (backward). They can store 30 minutes of video in any sized time period, with additional capacity reserved for recording accidents. The camera will be equipped with an “accident” sensor (gyroscope and impact sensor). When this sensor raises an alert, the camera starts recording an “accident” report (thus terminating any ongoing recording) on both the forward and backwards lenses simultaneously at half the frame rate (so the same rate of accumulation, but two streams) for three minutes per event.

The specification for the camera includes:

- HD video: 1280x720 pixels, 30 frames per second (fps), emergency recording is two streams at 15 fps each.
- Communications when the bike is not docked: the camera is wired to the tiny onboard processor, so its data transmission utilizes the communications of that processor. You will be deciding whether this is Bluetooth, cellular data, or Wifi.
- Communications when the bike is docked: In this case, the processor uses the bike “plug” supporting USB 3.1 (600MB/s transfer rate) and communicates with the station system and from there into the rider’s account in the central facility using the high speed wired network.
- Battery capacity: 45 minutes of recording time with a maximum of 30 minutes for rider recordings and the remainder of the time reserved for emergency recording.
- With compression on the camera, each frame is 1.5MB (with an accumulation rate 45 MB/s)
- Total storage is 128GB with 35 GB reserved for accident video recording

Bikes are equipped with a small computer and screen. For bikes that have a camera, the screen should indicate whether all video has been transferred.

4.2 Bike Stations

Each bike station consists of several docks, the bikes in the docks, and a kiosk, all supported by a small computer.

4.2.1 The station as a whole

Each station is a separate subsystem. It supports 15 to 45 docks and the kiosk and has the following configuration:

- Intel Xeon single processor
- 16 GB memory
- 1 TB storage
- 1 Gbps network interface (symmetric, so 1 each way)
- Battery backup for 2 hours of operation of the whole station

4.2.2 Docks

In terms of the overall system, the docks are simply components of the station itself. Besides being a storage and locking facility for the bikes, each one will have a power source for charging any batteries on the bike (other than the LoRa system), and a plug carrying USB 3.1 for data between the bike and the station. The plug will allow for reporting the bike ID, any GPS information, bike health information as measured by the health monitoring system, and any video that needs to be transferred.

The plug also communicates with the various batteries on the bike including for the processor and camera. All batteries other than for the LoRa radio can be recharged through this plug, although for the camera and motor this may take time and may only fully recharge with significantly long stays such as overnight.

4.2.3 Kiosks

The interface on the kiosk consists of a touch screen, a keyboard, and a card reader for both credit/debit cards and bike cards. The screen can show a map of bike stations to anyone, without a card. To begin the process of borrowing a bike, a card (either credit/debit card or bike card) must be provided in the card slot. Both the touch screen and keyboard can be used for entering information into the system, such as a preferred bike ID, choices for reservations, etc.

4.3 The central computing facility

This computing facility will support computation, storage, and communication at the center of the whole system. It contains:

- Intel Xeon CPU Max Series processor with 56 cores
- 256 GB DRAM memory

- 100 TB storage
- 1 Tbps network capacity (symmetric, so 1 Tbps each way)

The central system is expected to handle the full load of connections to it from riders in the app, bikes, and the stations. It has the capacity to handle at least 5,000 simultaneous connections from the combination of the app, the kiosks, the docks, and the bikes.

This system will both keep track of all the management information, such as user accounts, individual bikes, as well as information needed to make long-range decisions about the (re)placement of docking stations, the numbers of bike docks in those stations, the number of bikes, etc. Your team will need to consider the storage of accident videos.

5. Summary of Your Task

Your team's job is to design the subsystem that supports the experimental video transfer module. This includes:

- Basic functionality for transferring videos from the bike to the central computing infrastructure, as well as storing the videos on the server.
- Dealing with edge cases such as a rider trying to reserve a bike that is currently transferring a video.
- Determining what functionality to support during different types of failures, and how.
- Considerations of data collection and privacy when it comes to storing videos.

Appendix: A partial list of interfaces

This is a sampling of the interfaces in the system. It should give you a sense of both the kinds of requests that parts of the system can make of each other and user interface requests. It is intentionally incomplete because the design of some of the requests is part of your challenge. Note that every element of the overall system has its own address, so these messages are all addressed to particular elements (e.g., a particular bike, dock, station, etc.). If no return values are specified, you can assume just a simple return. These are in no particular programming language. Square brackets ("[]") indicate a list.

It is incredibly unlikely that your subsystem design will utilize all of these API calls. They are meant to illustrate the functionality of the system; part of your job is determining the subset that is relevant to your design. At the same time, you may also find that you need to define additional API calls, or adjust some of the ones given here. Any adjustments or additions should be well justified.

Bike to station through dock (through USB 3.1 plug)

Function call	**Return value** (if any)	**Notes** (if any)
`check_in(bikeID)`		
`check_out(bikeID, riderID, rideID)`		Both rider and ride need IDs

Station through dock to bike (through USB 3.1 plug)

Function call	**Return value** (if any)	**Notes** (if any)
`assign_bike(riderID, rideId)`		
`get_status()`	`tire_status, brake_status, rider_video_to_send_flag, rideID accident_video_to_send_flag, rideID`	
`get_accident_video(riderID, rideID)`	`accident_video`	This function call is incomplete. You will need to complete it for your assignment. It also may not be used at all.

GPS to central system (using LoRa)

Function call	**Return value** (if any)	**Notes** (if any)
`report_gps_location(bikeID, number_of_reports, [list_of_gps_reports])`	Number of reports received	
`turn_listening_mode_on(bikeID)`		
`turn_listening_mode_off(bikeID)`		

Central system to GPS (using LoRa)

Function call	**Return value** (if any)	**Notes** (if any)
`get_status()`	`tire_status, brake_status, battery_levels, GPS_location`	All location information is in the form of latitude, longitude (floating point degrees) and a timestamp.
`lock_bike()`	`True` if locked successfully; `False` otherwise	Lock bike not in dock. Note that there may be other ways to lock/unlock a bike
`unlock_bike()`	`riderID, rideID` if unlocked successfully; `null` otherwise	Unlock bike not in dock. Note that there may be other ways to lock/unlock a bike

Bike to central system (using cellular data or WiFi)

Function call	**Return value** (if any)	**Notes** (if any)
`send_rider_video(bikeID, riderID, rideID, video)`	`rideID` if received successfully; `null` otherwise	This may not be exactly what you need; you are welcome to propose a variation of this API
`send_accident_video(bikeID, riderID, rideID, video)`	`rideID` if received successfully; `null` otherwise	This may not be exactly what you need; you are welcome to propose a variation of this API

Central system to bike (using cellular data or WiFi)

Function call	**Return value** (if any)	**Notes** (if any)
`get_status()`	`tire_status, brake_status, battery_levels, accident_video_available, GPS_location, station_if_docked`	

`lock_bike()`	`True` if locked successfully; `False` otherwise	Lock bike not in dock
`unlock_bike()`	`riderID, rideID` if unlocked successfully; `null` otherwise	Unlock bike not in dock
`get_accident_video()`	`riderID, rideID, video`	This may not be exactly what you need; you are welcome to propose a variation of this API

Central system to station

Function call	**Return value** (if any)	**Notes** (if any)
`reserve_bike(riderID, time_of_reservation)`	`reservation_ID` if successful; `null` otherwise (e.g., if reservation declined)	
`reserve_dock(riderID, time_of_reservation)`	`reservation_ID` if successful; `null` otherwise (e.g., if reservation declined)	

Station to central system

Function call	**Return value** (if any)	**Notes** (if any)
`reserve_bike(riderID, stationID, time_of_reservation)`	`reservation_ID` if successful; `null` otherwise (e.g., if reservation declined)	Reservation made at kiosk for another station
`reserve_dock(riderID, stationID, time_of_reservation)`	`reservation_ID` if successful; `null` otherwise (e.g., if reservation declined)	
`send_rider_video(bikeID, riderID, rideID, video)`	`rideID` if received successfully; `null` otherwise	This may not be exactly what you need; you are welcome to propose a variation of this API

`send_accident_video(bikeID, riderID, rideID, video)`	`rideID` if received successfully; `null` otherwise	This may not be exactly what you need; you are welcome to propose a variation of this API
`get_non_member_id (credit/debit_card_info)`	`riderID`	
`new_ride(bikeID, riderID, rideID)`		

App to bike (using Bluetooth)

Function call	**Return value** (if any)	**Notes** (if any)
`get_new_video_list()`	`list_of_unretrieved_videos`	
`get_video(videoID)`	`video`	

App to central system

Function call	**Return value** (if any)	**Notes** (if any)
`get_station_information()`	`stationID, bikes_available, docks_available`	
`login(riderID)`		
`borrow_bike(riderID, stationID, bikeID)`	`True` if successful; `False` otherwise (e.g., if declined)	Bike ID is optional, if not provided, bike will be assigned by the system
`report_bike_issues(riderID, bikeID, rideID, set_of_issues)`		
`reserve_bike(riderID, stationID, time_of_reservation)`	`reservation_ID` if successful; `null` otherwise (e.g., if reservation declined)	Reservation made at kiosk for another station
`reserve_dock(riderID, stationID, time_of_reservation)`	`reservation_ID` if successful; `null` otherwise (e.g., if reservation declined)	

`find_route(origin, destination, list_intermediate_stops)`	`route_information_for_mapping`	
`revise_route(current_route, route_deviations)`	`route_information_for_mapping`	All route information will be in the form of a series of location points (latitude and longitude as floating point degrees)
`get_video_list()`	`list_of_videos_on_server`	
`get_video(videoID)`	`video`	

Kiosk interface (Note that only some of these are available to non-members)

Function call	**Return value** (if any)	**Notes** (if any)
`get_station_information()`	`stationID, bikes_available, docks_available`	
`login(riderID)`		
`create_one_time_riderID(credit/debit_card_info)`	`riderID` if successful; `null` otherwise (e.g., if declined)	
`borrow_bike(riderID, stationID, bikeID)`	`True` if successful; `False` otherwise (e.g., if declined)	Bike ID is optional, if not provided, bike will be assigned by the system
`report_bike_issues(riderID, bikeID, rideID, set_of_issues)`		
`reserve_bike(riderID, stationID, time_of_reservation)`	`reservation_ID` if successful; `null` otherwise (e.g., if reservation declined)	Reservation made at kiosk for another station

`reserve_dock(riderID, stationID, time_of_reservation)`	`reservation_ID` if successful; `null` otherwise (e.g., if reservation declined)	