HT

Hypothetical Transportation

School Bus Logistics Management System: Evolution 4

Tyler Bletsch, Duke University

March 28, 2022

1 Introduction and Use Cases

Your employer, Hypothetical Transportation (HT), is a provider of school bus services to a handful of school districts in your area. They operate a fleet of buses to transport K-12 children to/from school as well as to special activities and events.

They currently use a mishmash of spreadsheets, text files, and google docs to track their students, buses, and routes, but this is cumbersome and error-prone, so they would like a unified system to replace this.

This system will serve the following use cases:

- Users with appropriate privilege will be able to manage school, student, and parent records, and to note the relationships among these.
- Users with appropriate privilege will be able to see the layout of student homes, group students into a route, and document said route as text.
- Optionally, software implementors may elect to include a feature to auto-suggest groupings of students into routes.
- Parents will be able to login to review the bus route for each of their student(s).
- Parents will be able to activate their account via email.
- Users with appropriate privilege will be able to contact parents via email globally, per school, or per bus route, including emailing bus stop information directly to parents.
- Bus routes will be able to be designed using a map-based interface rather than just plaintext.
- Optionally, software implementors may elect to include a feature to auto-generate reasonable bus stops.

- Parents will be able to see available stops within walking distance depicted on a map.
- *School staff* will be included as users of the system, and they will be able to manage students/parents/routes for select schools within their purview, but are prevented from affecting other schools.
- *Bus drivers* will be included as users of the system, and they will be able to generate a link to commercial mapping software (e.g., Google Maps) for routes, download a printable route roster, and generally have read-only access to system data.
- Administrators and school staff will be able to bulk import information in a standardized tabular format.
- To keep access segmented, the system will restrict user actions via a role-based permission system.
- IT operations staff will be able to restore the state of the system from a robust backup system using a clearly documented procedure.
- To track route activity and allow location tracking, drivers will log the start and stop of routes and the specific buses driven, and logs of this will be kept and can be reviewed.
- The system will communicate with an existing bus location tracking system in order to provide parents and students with location and, at the implementor's discretion, ETA information. Mobile devices will be usable to access this bus tracking data. (For our purposes, this GPS tracking will be simulated.)
- To allow them to track their bus live, students will be able to be granted limited access to the system.
- To simplify system semantics, and parent and non-parent accounts will be mutually exclusive.
- The team will develop a sales video to potentially market this software to other school transportation management companies.

2 Definitions

1. **Implementor**: Refers to you, the software developer. Items described as "at the implementor's discretion" or similar indicate free choices. However, "free choice" does not mean all choices are of equal merit; your overriding goal must be *software quality*.
2. **Unique**: Requirements may describe a given field (or combination of fields) as *unique*. This means that there may be at most one record with that value (or combination of values). Attempts to violate uniqueness should generate an error, unless otherwise specified in the requirements.
3. **User**: A person able to login to the system. May be an administrator , a driver, school staff, a student, or a parent. ~~Note: students are *not* users of the system.~~ Defined by fields:

- Email address: Serves as the login name for the user. Short text field. Unique (case insensitively), required.
- Full name: The user's name. Short text field. Required.
- Address: Parents only. The address of the parent and their student(s). Short text field. Required only ~~if an account has associated student(s)~~ for parent accounts, disallowed for other accounts . Assisted location input (req 1.12).
- Phone number: Parents only. A phone number for the parent. Short text field. Required only ~~if an account has associated student(s)~~ for parent accounts, optional for other accounts .
- Password: Stored credential info used to validate a password; must be stored securely (see req 1.5). Required, unless the account is awaiting link-based password creation.
- Role: A single role as described in definition 10. ~~Optional, absent for unprivileged users.~~ Required.
- Students: Parents only. A list of references to zero or more students for which this user is the parent/guardian (see definition 4). A student can be associated with only one parent user.
- Managed schools: For the *school staff* role only, a list of zero or schools which this user manages.

4. **Student**: A pupil who rides the bus system. ~~Not able to login to this system.~~ Students *may* be granted access to the system, meaning that the student would also serve as a user with student role as described in definition 3 above. This will not be the case for all students. Defined by fields:
    - Full name: The student's name. Short text field. Required.
    - Student ID number: Assigned by the school or district. Positive integer, optional. *Not* assumed to be unique, as the number may have been assigned by one of several districts/systems.
    - School: A reference to the student's school, described in definition 5. Required.
    - Route: A reference to a bus route. Optional.
5. **School**: A school served by the bus system. Defined by fields:
    - Name: The name of the school. Short text field. Required. Unique (case insensitively).
    - Address: The address of the school. Short text field. Required. Assisted location input (req 1.12).
    - Bus arrival time: The time of day by which all school buses should have arrived in the morning. Generally set a bit before classes start. Required.
    - Bus departure time: The time at which school buses depart the school in the afternoon. Generally set a bit after classes end. Required.
6. **Route**: A path that a given bus follows to collect or drop off students.
    - Name: A name for the route, such as "Cherry Elementary northwest 2". Short text field. Required.

- Students: A set of one or more students expected to use this route. A student can belong to only one route.
- School: A reference to the school to/from which students are transported.
- Route description: A long-form, multi-line text field in which the route is described. *Clarification*: In light of the system now understanding actual stop locations (see below), this field becomes more of a comment or description field. This doesn't change anything about the design of this field, just its expected use.
- Stops: A set of one or more stops (per definition 7), stored in the order they will be visited by the bus. The bus will follow the stops in one direction in the morning, and the opposite order in the afternoon.

7. **Stop**: A single waypoint at which one or more students will be picked up or dropped off.
    - Name: The stop may be given a name. Optional: if omitted by the user, then a stop number will be used.
    - Location: Where the stop is. Assisted location input (req 1.12), with the further note that it should be possible to specify locations that aren't exact street addresses by fine-tuning the location with a GUI-selected GPS location ("dropping a pin"). For example, many stops will be of the form "the corner of Street A and Street B", which doesn't correlate to a specific address.
    - Pickup time: The expected time of arrival of the bus at this stop in the morning (on the way *to* school). This field is not input, but rather derived from the drive-time of the route and the bus arrival time of its school.
    - Dropoff time: The expected time of arrival of the bus at this stop in the afternoon (on the way *from* school). This field is not input, but rather derived from the drive-time of the route and the bus departure time of its school.
8. **In-range stop**: Legally, a stop must be within *three-tenths of a mile* (1584 ft.) of a student's home for it to be considered a feasible bus stop for that student. Such stops are said to be *in-range* for that student. This is computed as straight-line distance ("as the crow flies"). Note: a student may be in-range of more than one stop; there is no concept of an "assigned" stop.
9. **Complete route**: A route is said to be *complete* if all of its students have a stop that is *in-range*. Routes that do not have this property are said to be *incomplete*.
10. **Role**: Confers a set of abilities on the system for a user. ~~A user may not have multiple roles.~~ A user has exactly one role. Users role is assigned by users in the administrator role. ~~All users can access the parent features if they have associated students, so no explicit "parent" role exists.~~ Permissions associated with these roles are summarized in a table at the end of this document. The grantable roles are:
    - **Driver**: Allows the user to review all route information, including student rosters. Because a driver may need to substitute for another driver with little notice, drivers do not have an assigned route in this system and instead have broad read-only access. Can also start/stop bus runs.

- **School staff**: Allows the user to manage the parents, students, and routes associated with their managed school(s).
- **Administrator**: Can create and manage schools, parents, students, and routes. Can also confer or revoke a role onto users (per req 1.13).
- **Parent**: Able to have associated students and able to access a restricted version of the interface only showing data pertaining to those students.
- **Student**: Only able to access a restricted version of the interface only their own route/bus information. Not all students are users of the system.

11. **Administrator**: A user with the administrator role.
12. **Appropriate school staff**: This refers to a user with the school staff role who has the school in question among their managed schools. For example, if a requirement says "administrators or appropriate school staff" can create a route, the school staff member is implicitly restricted to doing so only for a school they manage.
13. **Unprivileged users**: ~~Users with no explicitly assigned role. It's expected that most parent accounts will fall into this category.~~ A shorthand term for parent and student accounts.
14. **Privileged users**: A shorthand term for administrator, school staff, and driver accounts.
15. **TranzitTraq system**: A GPS-based bus tracking system currently used manually by Hypothetical Transportation to monitor bus traffic. Created by the now-defunct TranzitTraq Solutions (out of business since 2009), little documentation remains. For the purposes of your software (and to avoid all the logistical headaches of a real bus GPS system), you will integrate with a TranzitTraq simulator rather than the real thing.
    - The URL for this system is http://tranzit.colab.duke.edu:8000/.
    - This URL will take you to a map interface where you can set or check the location of buses by bus number. For the purposes of keeping teams separate, we will assume that group X will only deal with buses numbered X000 through X999. So group 3 would only use bus numbers 3000 through 3999. Groups should never mess with numbers outside their range. Buses numbered less than 1000 are free for scratch testing.
    - Bus locations can be retrieved at http://tranzit.colab.duke.edu:8000/get?bus=X where X is the bus number. A self-explanatory JSON response will show the bus location (if set), or an error message if that bus's location has never been set in the interface.
    - Note: From time to time, you may find the system non-responsive or otherwise unreliable. Coping with this is your responsibility.
    - Meta-note: TranzitTraq Solutions is fictional, but companies using old, unsupported, but functioning gear is very much real. I did my best to give TranzitTraq that early 2000's web app feel. To aid with debugging, the raw HTTP access log is available via a link at the bottom of the site.
16. **Bus number**: An integer between 1 and 99999 identifying a physical school bus. A bus with a given number can only be on one run at a time.[1]

[1] Added 2022-03-27.

17. **Bus run** and **in transit**: A *bus run* is the act of driving a bus along a route, either picking up or dropping off students. A driver begins a bus run by registering a particular bus number and route with the system (req 10.2), at which time that bus and route are said to be *in transit* (e.g., "bus 909 is in transit on route 12, driven by Dave"). The run ends when the driver signals such to the system or after a three-hour timeout (req 10.3). A driver logging a bus as in transit of a given route is how the system learns what bus to associate with what route for location tracking purposes.

3 Requirements

A note on requirements: No set of requirements is perfect, and that is certainly true here. I'm sure that contradictions, under-specified behavior, and unintended consequences will be revealed. Your overriding goal should be to produce a quality system; if you believe that goal would be better served if a requirement were altered or interpreted a certain way, ask about it, and get the conclusion in writing. The result may be a variance in a requirement for a specific team, or even modification of this requirements document for all teams. In short, if unsure, ask.

Some requirements have attached an informal tip, motivation, or example; these do not alter the requirements themselves, but are meant to answer likely questions about a requirement.

1. **Server**

   1.1. Your software must have a server that supports an arbitrary number of users. A web-based solution is preferred; thick client or mobile options are available with instructor pre-approval only.

   1.2. During the install/setup process, at least one administrator user is configured. For test purposes, one such user should be "admin@example.com".

   1.3. Users must have their accounts created by an administrator before being able to use the system (see req 3.1 for account creation).

   1.4. A user accessing the system prior to logging in should be able to access nothing but a login prompt. Login is via email address and password.

   1.5. Any stored passwords must be kept in a secure manner (i.e., salted and hashed at minimum)

   1.6. Users may change their own password using the customary two-matching-blinded-inputs approach commonly seen.

      1.6.1. Users should be able to make use of a "forgot my password" facility, which uses email to allow a password reset. Note: the user's password should only be reset if they successfully complete the full process; merely doing the first step of choosing "forgot my password" should not invalidate credentials or affect the account.

   1.7. All communication between the clients and server must be encrypted.
   Tip: For web-based solutions, this means using HTTPS.

   1.8. The server must maintain state in a persistent fashion.
   Tip: For web-based solutions, this just means using a database or similar.

1.9. **Pagination rule**: For all views which show a potentially unbounded number of records, the response time of the interface shall not depend on the quantity of records unless a full listing is explicitly requested by the user.
Tip: This implies some form of pagination so that only a finite number of records are retrieved at a time. Pagination can be explicit (page 1 of N) or implicit (infinite scrolling with auto-loading). The latter part of the requirement ("unless a full listing is requested") implies a "show all" button or similar. Other UI solutions are likely also possible.

1.10. **Consistency rule**: A variety of cross-references are made by the system; the system must maintain internal consistency of these references in all cases. For example: a student belongs to a school, a bus route is associated with that school, and that student belongs to that bus route, therefore, both the route and the student must be associated with the same school.

1.11. **Assisted selection**: A user input is said to be *assisted* if it is a user-selected reference to an existing record (student, school, user, etc.) where the UI provides a listing, inline search, autocomplete, and/or other means to allow easy and efficient selection. Unless otherwise specified in this document, all selections of an existing record should be assisted.

1.12. **Assisted location input**: A location selection is said to be *assisted* if, after specifying the address, some form of map integration is used to verify the addressed was understood, correlate it to a specific location on Earth, and confirm this via some form of map display. Implementors may even opt to allow the user to use a GUI to fine-tune the coordinate of the location in question. Unless otherwise specified, all situations that require input of a location should be assisted in this way.

1.13. The system shall track the role for each user as described in definition 10; a user may not do anything outside the capabilities described in their role.
Note: Every effort has been made so that these requirements reflect the roles described. Please notify the instructor of any discrepancies found. If a requirement says a certain role can do something and a sub-requirement simply says that the "user" will do part of that thing, then the word "user" in that context is simply a pronoun referring to the initiating user who has the appropriate role, not just any normal user.

2. **School record management**

2.1. **School create**: Administrators will be able to create schools, recording for them all fields specified in definition 5.

2.2. **School list**: Administrators, drivers, and appropriate school staff will be able to view a table of schools with all fields. For school staff, the list will be limited to schools the user manages.

2.2.1. The view should be sortable by school name, bus arrival time, and bus departure time.

2.2.2. It should be possible to filter this view by keyword search on the school name.

2.2.3. Users should be able to navigate from this to a detail view for a school (see req 2.3).

2.3. **School detail**: Administrators, drivers, and appropriate school staff may view a detail view of a school showing all fields as well as a listing of the associated bus routes and students. Students with no bus route or who are out-of-range of all their route's stops should be shown distinctively. Routes that are incomplete should also be shown distinctively. It should be possible to navigate to detail views for associated routes (req 4.3) and students (req 3.7) from here. It should be possible for administrators and appropriate school staff to modify the school from here (req 2.4), and for administrators (not school staff) to delete the school from here (req 2.5). Finally, it should be possible for administrators and appropriate school staff to navigate to the route planner interface from here (req 4.1). A link or interface to send school-wide emails should be available from this view (see req 7) for administrators and appropriate school staff. It should be possible to view or navigate to a transit status map for buses on routes for this school as in req 10.6.

2.4. **School modify**: Administrators and appropriate school staff will be able to modify school records. Changes to the a school's bus arrival/departure times should automatically update associated route bus stop pickup/dropoff times, as they are derived values. School staff may not modify school name or address.

2.5. **School delete**: Administrators will be able to delete schools. As this is rare, a highly visible confirmation shall be required to do this in which the administrator must confirm the name of the school by typing it (case insensitive). Upon deleting a school, all its students and routes should also be deleted, but associated parent accounts will remain.

3. User and student record management

3.1. **User/student create**: Administrators will be able to create parent accounts and their associated student(s) (if any) in an efficient dialog, recording for them all fields specified in definitions 3 and 4. This interface or a similar one will ~~also double~~ serve as the means to create new administrator and other ~~role-bearing~~ privileged accounts. School staff will also be able to create parents accounts and students, associating such students only with school(s) among those managed by the school staff user. School staff may only create parent and student accounts, not ~~role-bearing~~ privileged accounts.

3.1.1. In specifying the parent user account, it should be possible to refer to an existing parent account (i.e., using the interface to add a student to an existing parent) or to create a new parent user account inline (adding parent+student at the same time). If the user undertaking this action is school staff and they input a parent email address already associated with an existing account, they should notified of that fact and allowed to add students to that existing parent account. This is true even if they previously couldn't view that parent because they previously had no students enrolled at one of the user's managed schools.

3.1.2. To allow for secure account creation, the implementor has a choice of implementing either a *one-time password* or *link-based password creation*. In the case of a one-time password, the account is given a randomly generated password by the system that must be changed by the user upon their first login. In the case of link-based password creation, a special URL is provided to the user which, when visited, will prompt the creation of the account's password; once a password is created in this way, the special URL should no longer work. In either case, the process will begin with an email being sent to the user explaining the system and providing the one-time password or special URL.

3.1.3. ~~Specifying a student is optional, as this interface is also a means to create new administrator or other role-bearing accounts. However, note that administrator accounts and other role-bearing accounts *can* have associated students, as such people may have kids.~~ If a parent account is being created, associated students can be created at this time as well. The user should be able to elect to associate such students with login accounts at their discretion, filling out the additional needed detail (email address) only in that case. Student accounts created in this manner should trigger the same email-based password creation process as other accounts.
Rationale: Young children would not be issued accounts, but older middle and high school students may.

3.2. **User list**: Administrators and drivers will be able to view a table of all users showing all fields (except password, of course). School staff may also view such a list, but it will be filtered to just accounts with students enrolled at schools among those managed by the user.

3.2.1. The view should be sortable by email address and name.

3.2.2. It should be possible to filter this view by keyword search on the name or email address, and it should be possible to filter by role as well.

3.2.3. Users should be able to navigate from this to a detail view for a user (see req 3.3).

3.3. **User detail**: Administrators, drivers, and appropriate school staff may view a detail view of a user showing all fields as well as a listing of the associated students (if any). School staff should only see students attending a school they manage. Students with no bus route or who are out-of-range of all their route's stops should be shown distinctively. It should be possible to navigate to detail views for such students (req 3.7) from here. It should be possible to modify or delete the user from here (reqs 3.4 and 3.5) if such actions are permitted. If the user is a driver currently in transit, then this should be indicated, the bus number and route shown, and a link to the route's detail view (req 4.3) made available.

3.4. **User modify**: Administrators and appropriate school staff will be able to modify user accounts. Administrators should *not* be able to reset passwords; such a facility is superfluous given the self-service password reset in req 1.6.1. Administrators can ~~grant or revoke a role to any existing user~~ change the role of any priviliged user , including the administrator role. An administrator may not revoke their own administrator role, however. ~~Accounts created with the parent or student roles cannot have their roles changed.~~Accounts cannot have their role changed to or from student or parent.[2] . Administrators may add or remove schools from a school staff account's managed school list. School staff are prevented from changing roles on a user or adjusting the managed schools of any school staff account.

3.5. **User delete**: Administrators and appropriate school staff will be able to delete users. A clear confirmation dialog should be shown first. Upon deleting a user, all associated

[2] Added 2022-03-27.

students are also deleted. School staff will be limited to deleting ~~unprivileged~~ parent accounts whose students are *exclusively* enrolled schools among those managed by the user. In other words, school staff cannot remove a user that is a parent to student(s) not enrolled at one of their managed schools; if this is attempted, a clear error message should be shown. Administrators and appropriate school staff may delete student accounts; this may be done in a manner that preserves the student record (merely revoking that student's ability to login) or as part of overall student record deletion (req 3.9).

3.6. **Student list**: Administrators, drivers, and appropriate school staff will be able to view a table of students showing all fields, as well as the name and phone number of the student's parent , as well as email address (for student equipped with login accounts) . Administrators and drivers may see all students, while school staff are limited to those attending a school the user manages. Students with no bus route or who are out-of-range of all their route's stops should be shown distinctively.

3.6.1. The view should be sortable by student name, student ID number, email address, and school.

3.6.2. It should be possible to filter this view by keyword search on the student name , email address, or student ID number.

3.6.3. Users should be able to navigate from this to a detail view for a student (see req 3.7).

3.7. **Student detail**: Administrators, drivers, and appropriate school staff may view a detail view of a student showing all fields (including email address for student equipped with login accounts) as well as a listing of the associated route and school. Parent contact info (name, email, home address, phone number) should also be shown here. It should be possible to navigate to detail views for route (req 4.3) and school (req 2.3) from here. It should be possible to modify or delete the student from here (reqs 3.8 and 3.9) if such actions are permitted. If a student has no route or if the student is out-of-range of all their route's stops, this should be made clear. If a bus is in transit on this student's route (req 10), this should be made clear, the bus number and driver involved should be shown, and its location should be shown on the map. If ETA tracking is supported by the system, the ETA to this student's in-range stops should be shown as well (req 10.5).

3.8. **Student modify**: Administrators and appropriate school staff will be able to modify student records, including modifying any field and associating the student with a different parent account or with a different school. If a school staff member is associating a student with a different school, it must be one managed by the staff member. It should also be possible to add an email address, thus kicking off an account creation process for the student, or to remove such an email address, removing the student's ability to login.

3.9. **Student delete**: Administrators and appropriate school staff will be able to delete students. A clear confirmation dialog should be shown first. Deleting a student will delete their associated student login account, if present.

4. **Bus route management**

4.1. **Route create**: Administrators and appropriate school staff will be able to create and maintain routes as follows:

4.1.1. From the school detail view (req 2.3), the user will navigate to a route planner for that school.

- 4.1.2. A map view of the school and all student home locations will be shown. The school will be shown distinctively, and students without current bus routes will be shown distinctively as well. At the implementor's discretion, students may be visually coded as to which route they belong to.
- 4.1.3. The user will group students visually into separate routes. Implementors have broad discretion as to how exactly to do this, with possible interfaces including click-to-color, a mouse-driven shape selection tool, and many more.
- 4.1.4. **Student auto-grouping** (*extra credit*): Optionally, implementors may develop an algorithm to automatically suggest groupings that would form the basis of a reasonable route. The algorithm should take into account student locations to recommend a grouping likely to minimize bus drivetime, and there are many possible approaches. When applied, this algorithm should only affect students not already associated with a route, and after the algorithm is applied, the user should still be able to edit the groupings as in req 4.1.3. If this feature is implemented, it is up to the implementor to document this in the Feature Guide (req 6.3) and demonstrate it during the evaluation session.
- 4.1.5. For each set of students grouped into a route, the user can name the route and write a long-form written description of it.
- 4.1.6. The user will then be able to specify stops by interacting with some form of mapping interface. This includes specifying the stop names and locations as well as the ordering of the stops. The interface should make clear when stops are in-range of students' homes and when the route becomes complete (servicing all students associated with it). The interface should allow flexible editing and revision of the route rather than just rigid sequential input (e.g., by allowing the re-ordering of stops).
- 4.1.7. **Stop auto-creation** (*extra credit*): Optionally, implementors may develop an algorithm to automatically suggest a sequence of stops. The algorithm should take into account student locations to recommend a route likely to minimize bus drivetime, and there are many possible approaches. When the feature is requested, it should only add stops to out-of-range students and/or prompt the user before overwriting existing stops. After the algorithm is applied, the user should still be able to edit the stops. If this feature is implemented, it is up to the implementor to document this in the Feature Guide (req 6.3) and demonstrate it during the evaluation session.

4.2. **Route list**: Administrators, drivers, and appropriate school staff will be able to view a table of routes showing their names, associated school, and number of students. School staff are limited to seeing routes associated with schools they manage, others can see all routes. Routes that are in transit should be displayed distinctively, and the driver and bus number shown. Routes that are incomplete should be shown in a distinctive way.

- 4.2.1. The view should be sortable by name, school, and student count.
- 4.2.2. It should be possible to filter this view by keyword search on the route name.
- 4.2.3. Users should be able to navigate from this to a detail view for a route (see req 4.3).

4.3. **Route detail**: Administrators, drivers, and appropriate school staff may view a detail view of a route showing all fields as well as a listing of the associated students (if any), as well as a map showing their locations, the stop locations, and the location of the school. If a bus is in transit on this route (req 10), this should be made clear, the bus number involved should be shown, and its location should be shown on the map. A stop time-table should also be shown with stop names and expected dropoff/pickup times; this should include the school itself with its bus arrival/departure times. It should be possible to navigate to detail views for such students (req 3.7) and the school (req 2.3) from here. It should be possible to modify or delete the route from here (reqs 4.4 and 4.5) if such actions are permitted. If the route is incomplete, this should be made clear, with the out-of-range students shown distinctively. A link or interface to send route-wide emails should be available from this view (see req 7) if this action is permitted. From this view, it should be possible to view or navigate to a transit log for this route (req 10.7).

4.3.1. **Navigation link**: From this view, a hyperlink to a commercial navigation service (e.g., Google Maps) should be provided which gives driving directions that follow the bus route in either morning (to-school) or afternoon (from-school) order. The intent is for this to be used with the driver's smart phone, so while the software overall does not need to be mobile friendly, the pathway from login to tapping this navigation link (and other driver features per req 10.1) should be achievable on an average smart phone.

4.3.2. **Printable route roster**: From this view, it should be possible to generate a printable roster of students on this route. It should be headed by the route name and school name, with the body being a table of all students, including the student's name, student ID, home address, parent name, parent email, and parent phone number. Here, "printable" means either an HTML output that prints nicely without extraneous UI elements or a PDF. All content should be black-on-white.

4.4. **Route modify**: Administrators and appropriate school staff will be able to modify routes. For the text fields (name, route description) , this is a simple form. For the associated students and stops, the graphical selection interface from req 4.1 is to be used.

4.5. **Route delete**: Administrators and appropriate school staff will be able to delete routes. A clear confirmation dialog should be shown first. Upon deleting a route, all associated students will revert to having no bus route. Implementors have their choice of deleting or preserving transit log entries associated with the deleted route.

5. **Parent and student interface**

5.1. Any parent user with associated students will, upon logging in, see a list of their student(s). From this list, the user can navigate to a view showing the student's name, student ID number, school, and bus route information: the route name, description, and a list and map of stop(s) that are in-range along with their dropoff/pickup times. If a bus is in transit on a student's route, its location should be shown (req 10). If ETA tracking is supported by the system, the ETA to this student's in-range stops should be shown as well (req 10.5). This interface should be reasonably accessible via mobile device.

5.2. ~~Unprivileged users~~ Parent users should *only* be able to see information about their student(s) and the associated route for each. Similarly, students should only be able to see their own information. By no means should such a user be able to see information about other students, details of other routes, etc. Further, the system is fully read-only for such users.

5.3. Any student user will, upon logging in, see their student ID number, school, and bus route information: the route name, description, and a list and map of stop(s) that are in-range along with their dropoff/pickup times. If a bus is in transit on the student's route, its location should be shown (req 10). If ETA tracking is supported by the system, the ETA to the student's in-range stops should be shown as well (req 10.5). This interface should be reasonably accessible via mobile device.

6. **Documentation**

6.1. **Developer guide**: A document shall be provided which orients a new developer to how your system is constructed at a high level, what technologies are in use, how to configure a development/build environment, and how the database schema (or equivalent) is laid out.

6.2. **Deployment guide**: A document shall be provided which describes how to install your software entirely from scratch. It should start by describing the platform prerequisites (e.g., Linux distro, required packages, etc.), then mechanically describe every step to deploying your system to production readiness.

6.2.1. In addition to covering how to install the system with "stock" default data, the procedure to install the system from scratch using backed up data should also be included (i.e., disaster recovery).

6.3. **Feature guide**: Optional. If an extra credit requirement is pursued, document its design, benefits, and a walkthrough of how to demonstrate it here. If you pursued extra credit in an earlier evolution, maintain its documentation, modifying it if needed.

6.4. **Backup admin guide**: A document shall be provided which explains the backup solution so that a system administrator unfamiliar with your software could configure it from scratch, restore the database to any given backup, and test a backup for validity. See req 9.

7. **Communication facility**

7.1. **General announcement**: Administrators and appropriate school staff shall be able to send an email to (a) all parent[3] users of the system, (b) all users associated with students enrolled in a given school, or (c) all users associated with students on a given route. School staff will be limited to schools which they manage and their associated routes. Emails sent in this way will include a user-authored subject line and a user-authored multi-line body.

7.2. **Route announcement**: Administrators and appropriate school staff shall be able to direct the system to send a route announcement email to (a) all parent[4] users of the system, (b) all users associated with students enrolled in a given school, or (c) all users associated with students on a given route. School staff will be limited to schools

[3]Added 2022-03-27.
[4]Added 2022-03-27.

which they manage and their associated routes. Emails sent in this way will include a configurable user-authored subject line, a user-authored multi-line body, and the same information as in the parent interface (req 5.1), tailored to each specific parent and shown in text form.

7.3. For the content format, simple plaintext is acceptable. Support for HTML formatting, images, attachments, etc. are not required (though can be included at the implementor's discretion).

7.4. Emails sent from the system should not reveal users' email addresses to each other (i.e., user addresses should go in the BCC field).

7.5. Emails should appear to come from a "no reply" or system service email address , or from a dedicated address at a traditional email provider such as Gmail ; they should not appear to come from the administrator authoring them. There is no need for the system to be able to receive or respond to email.

7.6. For efficiency in testing, the software implementors may opt to have the system quietly skip sending email to any address ending in "@example.com". No other domain should be ignored in this way.

8. **Bulk import facility**

8.1. Administrators and appropriate school staff shall be able to import parents and students by means of a format compatible with modern spreadsheet software (CSV, XLSX, or similar). Users will be subject to the same constraints in importing records as with manually adding them (e.g., school staff are limited to making students associated with a school they manage). **Hypothetical Transportation is accepting proposals on the format.**

8.2. The import interface shall include documentation as to the import format.

8.3. For parent users, the fields to be imported are: email address, full name, address, phone number, and a number of associated students (see below). Password shall not be imported, and all accounts created in this way will be ~~unprivileged~~ parents .

8.4. For students, the fields to be imported are: full name, student ID number, and school. Email will be an optional field, the presence of which implies the creation of a student account for the given student record. The school name must match an existing school known to the system (though the reference is case-insensitive and whitespace-insensitive). Additionally, some method of correlating students and parents should be part of the import format.

8.5. **Bulk import validation**: Upon receipt of the bulk import data, the records to be imported will be shown so they can be reviewed by the user.

8.5.1. Addresses: User addresses are an *assisted location input* (req 1.12), so the user will be able to review address locations and correct any issues, including addresses which could not be geographically matched.

8.5.2. School matching: Student school must match an existing school (case- and whitespace-insensitively). Non-matched schools can be reviewed and corrected by the user.

- 8.5.3. Exclusion: Some records may be marked by the user as being excluded from the actual import (e.g. via a checkbox). Such records will be ignored entirely, and thus any errors involved in them are no longer relevant.
  Rationale: The user may elect to do this if a record requires manual research or if it is determined to be a duplicate (see below).
- 8.5.4. Duplicates: The system should attempt to identify duplicate records (case- and whitespace-insensitive matches on email address, user name, or student name). Such duplicates should be shown distinctively, a comparison to the existing similar record(s) should be shown, and such records should be marked for exclusion from the import by default.
- 8.5.5. Catch-all: Any other issue which would cause the actual import to fail should be shown with enough detail to allow the user to correct it.
- 8.5.6. Commit/cancel: Once the user has reviewed all of the above, they can elect to commit (proceed with actual data import) or cancel the operation. Either choice should require explicit confirmation.

8.6. Upon receiving the command to commit the import, if any error is encountered, the entire action will be aborted. If such issues arise, the precise nature of the error should be presented to the user in enough detail that it can be corrected in the import data. In other words, bulk import is an all-or-nothing transaction.

8.7. After a successful import, a confirmation with the count of users and students imported should be provided.

9. **Backups**: You must deploy a backup solution for your system's database.

9.1. Backups shall be automatic and taken daily.

9.2. Backups shall be kept with a staggered retention (7 daily backups, 4 weekly backups, 12 monthly backups).

9.3. Backups must be stored on a separate system.

9.4. The backup system must require separate credentials to access.

9.5. The backup system should report on progress and alert on failure; this could be via email or another directed communication mechanism.

9.6. The backup system may be built either out-of-band from the main software (e.g. a background database dump restored manually by a sysadmin) or in-band (e.g. the software itself exporting its database using internal automation). If it is in-band, all backup operations should be restricted to administrators.

10. Driver interface, location services, and transit logging

10.1. **Mobile device support**: Drivers should be able use a mobile mobile device to access a route navigation link (per req 4.3.1) and the ability to start/stop a run (reqs 10.2 and 10.3) on a given route. As both of these involve selecting a route, this implies some form of mobile-capable route listing, which may be the same as specified in req 4.2 or a separate mobile-only one at the implementor's discretion.

10.2. **Starting a run**: The system should allow a driver to indicate that they are starting a run of a given route, indicating that they are placing a bus in transit. To this end, the driver will choose a route, give a bus number, and indicate the direction (to-school or from-school). If this driver already on a run, a warning should be shown, and if confirmed, that run is stopped and the newly described run begins. Similarly, a warning requiring confirmation should be shown if the given bus is already on a run or if any bus is running this route, as a bus can only be on one run at a time and a route can only have one run going at a time.[5] The bus, driver, and route are now said to be in transit, and this event (user, bus number, route, date, time, and direction) is written to the transit log.

10.3. **Stopping a run**: If a driver is in transit, the system will allow them to indicate an end to the run. This event is written to the transit log. If the driver fails to log an end to the run within three hours of it starting, it is times out automatically, and this timeout is written to the transit log.

10.4. **TranzitTraq support**: While any bus is in transit, the system should communicate with the TranzitTraq system to ascertain bus location as needed. If a route is in transit, all maps concerned with that route should show the current bus location. The system should gracefully cope with any failures, errors, and irregularities of the TranzitTraq system. The system should make no more than 100 requests in any 10-second period for an average request rate limit of 10 requests/second.[6]

10.5. **ETA support** (*extra credit*): Optionally, implementors may develop an algorithm to compute and present an estimated time of arrival (ETA) of a given bus to a student's in-range stops; this is to be available in a student's self-service view (req 5.3), a parent's view of the student (req 5.1), or a privileged user that's viewing a student's detail view (req 3.7). In implementing this feature, there are many possible approaches of differing complexity, accuracy, and robustness. For example, using a mapping API to find drivetime from current location to the stop is simple, but ignores the route's path, whereas taking the full route into account would involve inferring the "current" stop based on travel up to this point, adding complexity. If this feature is implemented, it is up to the implementor to document this in the Feature Guide (req 6.3) and demonstrate it during the evaluation session.

10.6. **Transit status map**: It should be possible for administrators, drivers, and appropriate school staff to view a live-updating map of all buses in transit. School staff will be limited to buses relating to schools they manage. It should be possible to review bus numbers, identify drivers, and link to their routes from here.

10.7. **Transit log**: It should be possible for administrators, drivers, and appropriate school staff to view a history of bus runs as inferred by start/stop events. This tabular view should show driver, bus number, school, route, direction (to or from school), start date/time, and duration (or "ongoing" for runs still in transit). School staff should only see records relating to schools they manage. Drivers, schools, and routes should link to their respective detail views (reqs 3.3, 2.3, 4.3). The view should be sortable by all fields and filterable by bus number, driver, school, and route.

[5] Reworded 2022-03-28 to clarify: max one run per bus and one run per route.

[6] Added 2022-03-27.

11. **Marketing video**

    11.1. Your company's leadership has decided that it may be feasible to market this software to additional districts and transportation companies. As your company is a small start-up, you do not have a formal marketing team, so your group has been asked to develop a 4-6 minute sales video to kickstart your effort.

    11.2. Points will be awarded for professionalism, succinctly capturing your value proposition, clearly differentiating from competitors, and overall attractiveness of visual aesthetic. Some extra credit points will be available; these will be awarded competitively.

    11.3. Submission of this component will be via YouTube link (either public or unlisted) submitted by a means to be announced separately.

Entity	**Action**	**Admin**	**School Staff**	**Driver**	**Parent**	**Student**
School	**Create/Delete**	y	-	-	-	-
School	**Modify**	y	M (times only)	-	-	-
School	**Read**	y	M	y	-	-
User	**Create/Delete**	y	M (non-priv only)	-	-	-
User	**Modify**	y	M (non-priv only)	-	-	-
User	**Read**	y	M	y	-	-
Student	**Create/Delete**	y	M	-	-	-
Student	**Modify**	y	M	-	-	-
Student	**Read**	y	M	y	Theirs only	Themself only
Route	**Create/Delete**	y	M	-	-	-
Route	**Modify**	y	M	-	-	-
Route	**Read**	y	M	y	Their students' only	Theirs only
Route	**Start/stop run**	-	-	y	-	-
Email	**All**	y	-	-	-	-
Email	**School**	y	M	-	-	-
Email	**Route**	y	M	-	-	-

Table 1: This table summarizes permissions associated with each role. The abbreviation "y" means yes and "M" means "for managed schools only".