**Connected Campus Crew**

Mark Girguis, Kishan Patel, Ankit Monga,

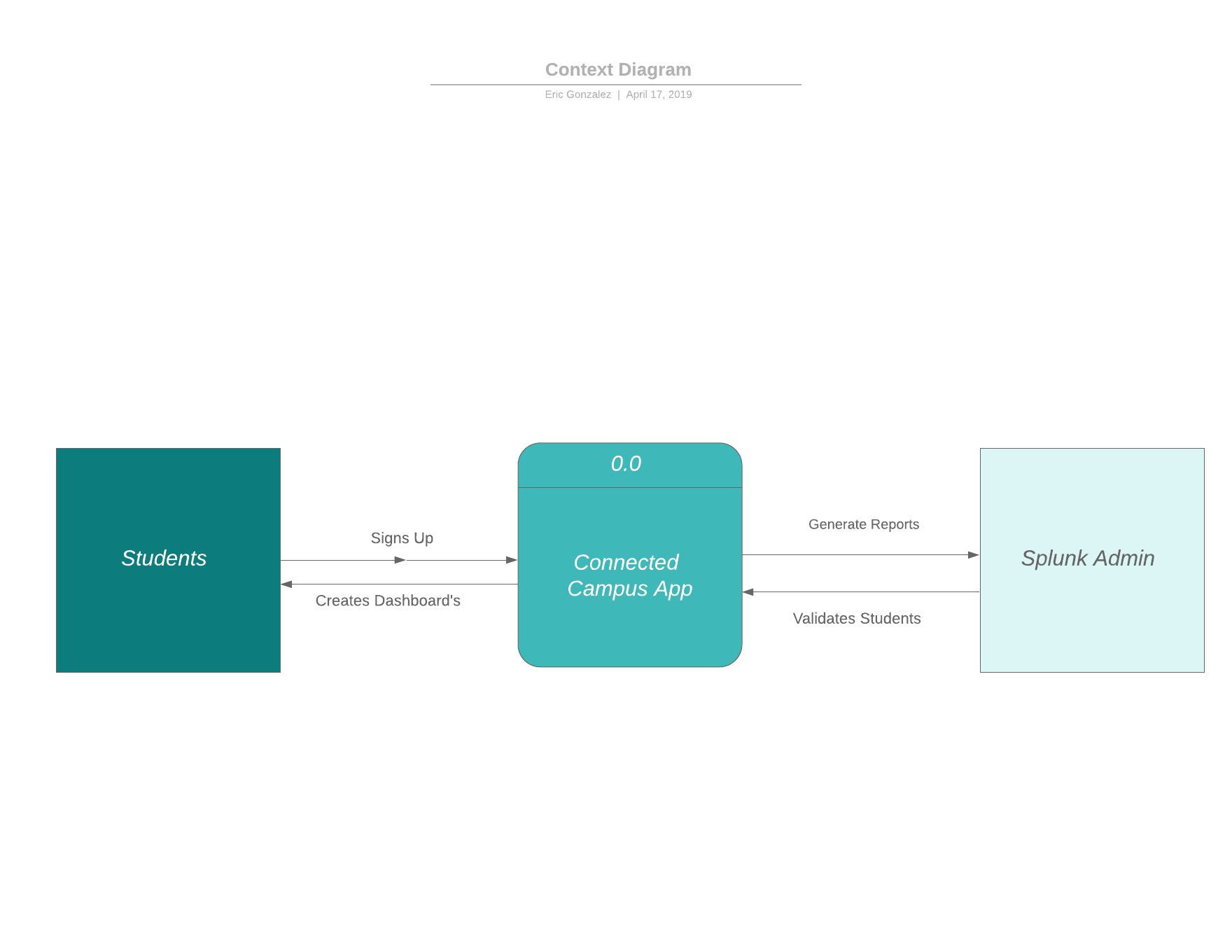
Eric Gonzalez, Jason Chung, Hector Khuon

**Deliverable 3**

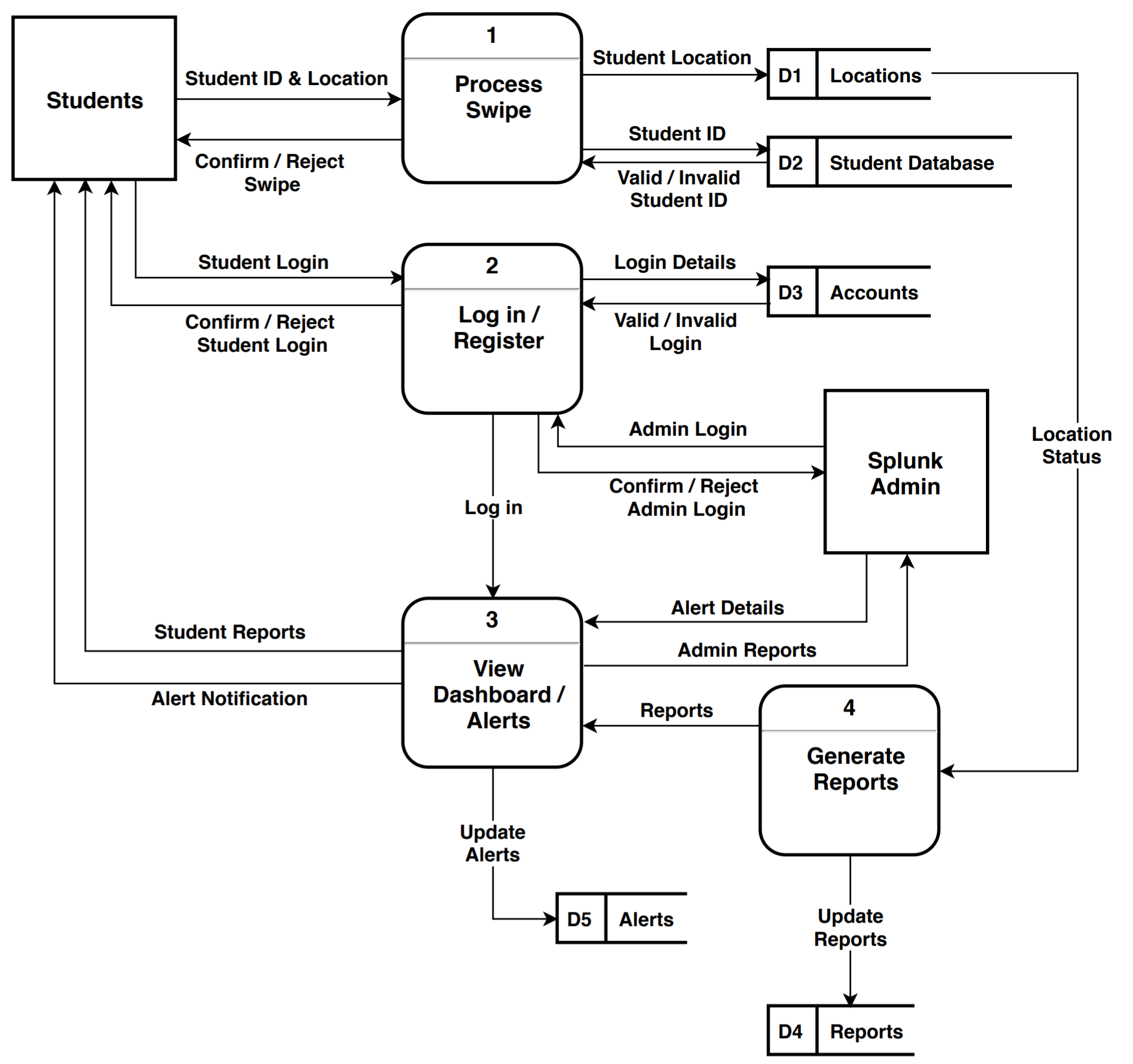
**IS 436 - 03**

**4/18/19**

**Context Diagram:**

****

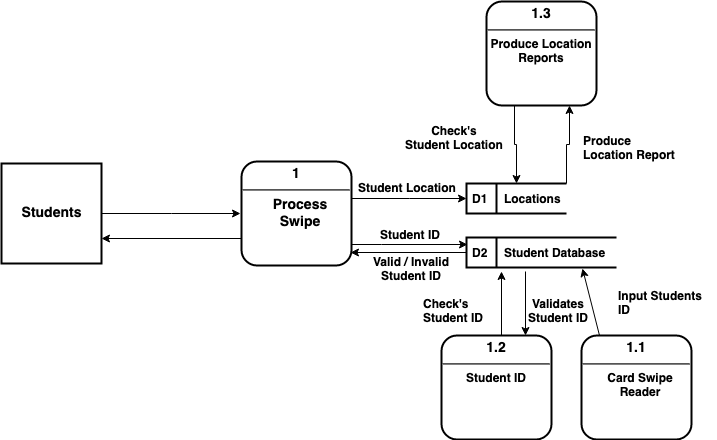
**Level 0 Diagram:**

****

Level 0-n explanation: Student logs into database, user searches traffic information, traffic information is searched and retrieved from the database, the database gives the user the requested traffic info.

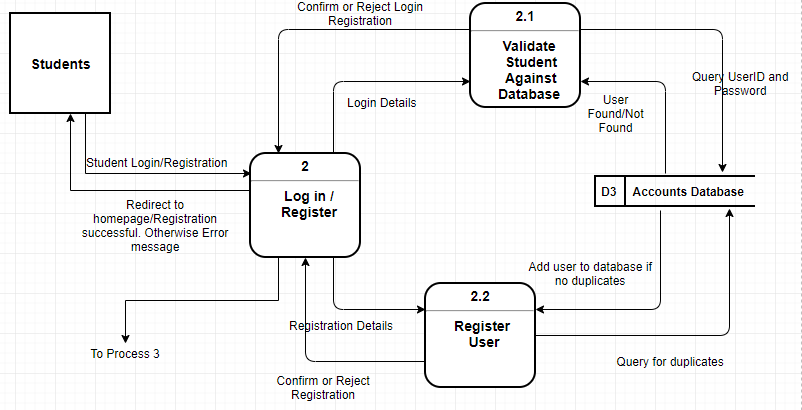
**Level 1 Diagram Process 1:**

**Process Swipe**

****

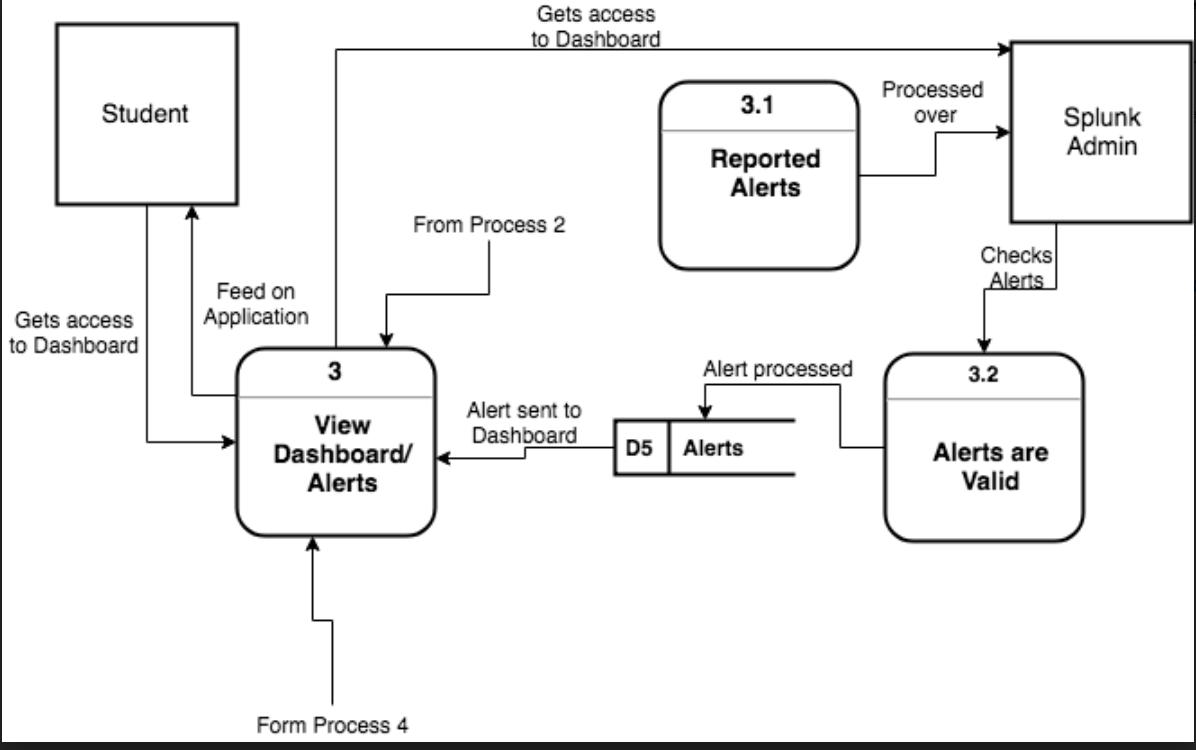
**Level 1 Diagram Process 2:**

**Login/Registration**

****

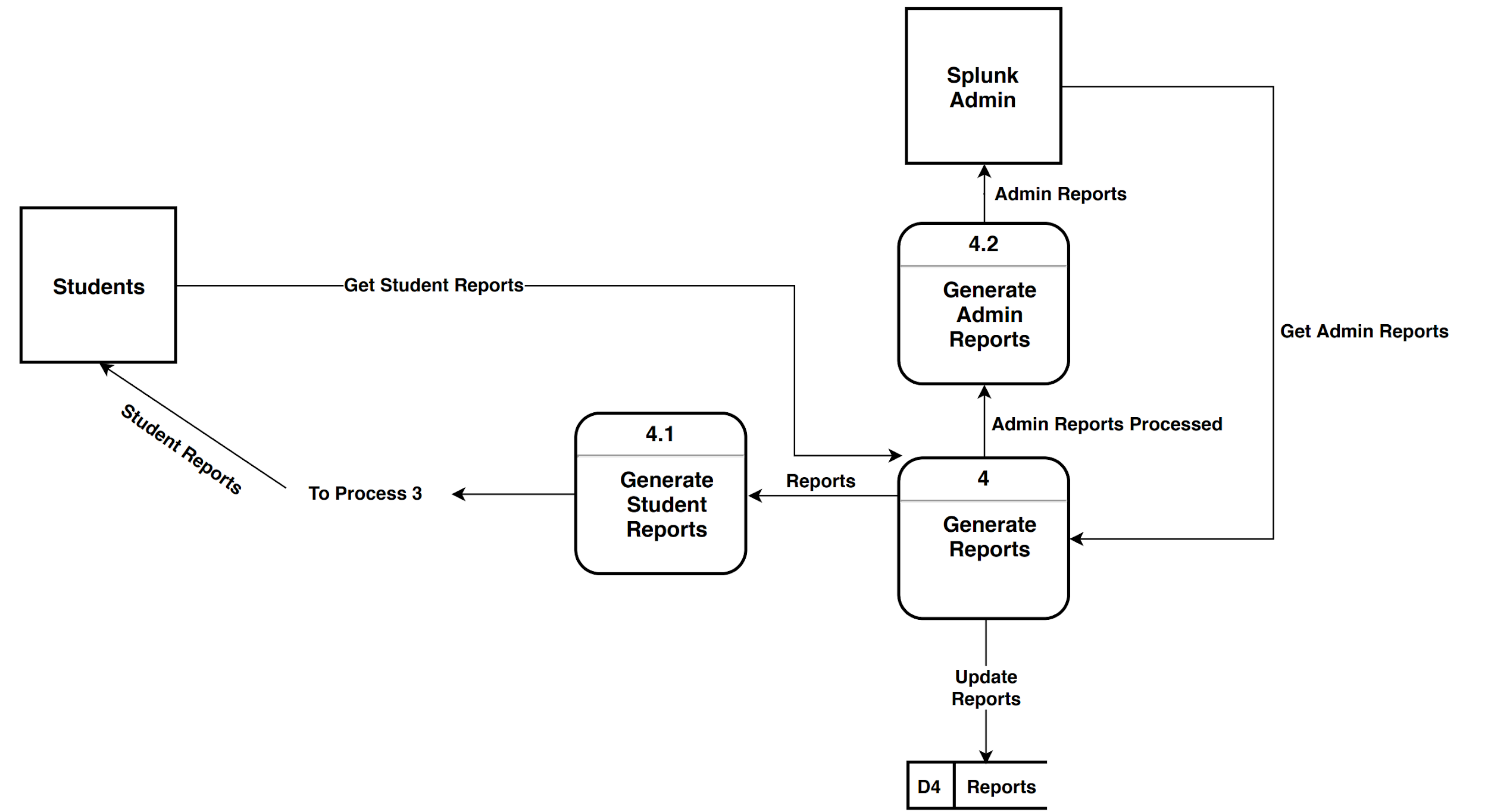
**Level 1 Diagram Process 3:**

**View Dashboard/ Alerts**

****

**Level 1 Diagram Process 4:**

**Generate Reports**

****

**Requirements Definition:**

Business/User Requirements:

Using this system as a Web-based application, UMBC (specifically DoIT) will have access to dashboards of student activity on campus in real-time. The specific functionality that the system will have will include:

* Alerts that would be triggered once an area gets overpopulated, to inform users in the area about high-traffic
* Alerts that would be triggered if a facilities issue occurs
* Option to make the alert located to a geographic location (using WAP data) or to trigger a broadcast alert to every student in the database
* Posting Alerts to Organization website and Mobile Application.
* Ability to correlate student LMS engagement to grade distributions
* Permission-based interface
* Agile and flexible dashboard updating in real-time
* Easy to learn UI

Functional Requirements:

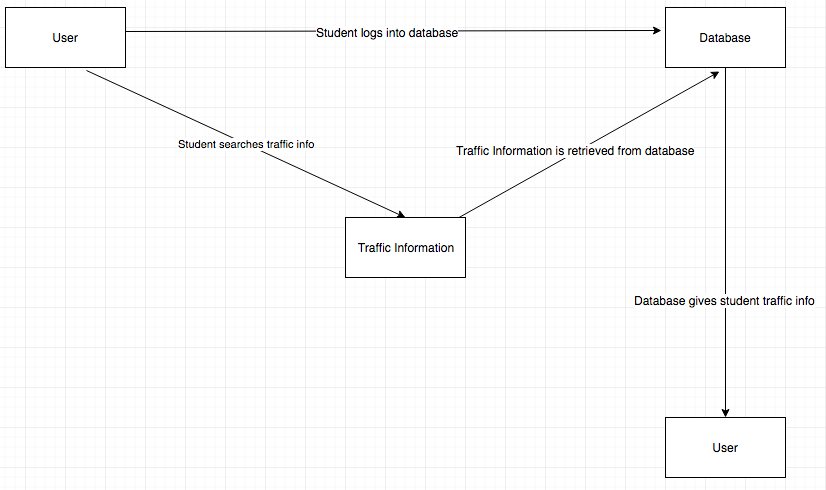
|  |  |  |
| --- | --- | --- |
| Functional Requirement | Description | Examples |
| Process-  Oriented | A process system must perform;  A process system must do | * The system must display an easy to learn UI where they can take analyze the data * The system must allow users to send alerts based on location |
| Information-Oriented | Information the system must contain | * The system must track students/faculty on campus from information obtained through the magnet swipe readers. * The system has a database of each individuals swipe from each respective building. |

Non-Functional Requirements:

|  |  |  |
| --- | --- | --- |
| Non-Functional Requirements | Description | Examples |
| Operational | The physical and technical environments in which the system will operate | * The system will run on a web-based application * The system will integrate existing magnet strip readers * The system will be compatible with multiple web browsers |
| Performance | The speed, capacity, and reliability of the system | * The data should be flowing in in real time * There should be little to no delay when a user swipes to the data being integrated into the UI |
| Security | Who has authorized access to the system under what circumstances | * Only admin will be able to have access to all the data * Students/ Faculty will not be able to view specific individual logs |
| Cultural and Political | Cultural and political factors and legal requirements that affect the system | * Personal information will be protected |

**Use Cases**

**Use Case 1(DFD):** Student Logs into database to see if the building they are visiting has high traffic at given time.

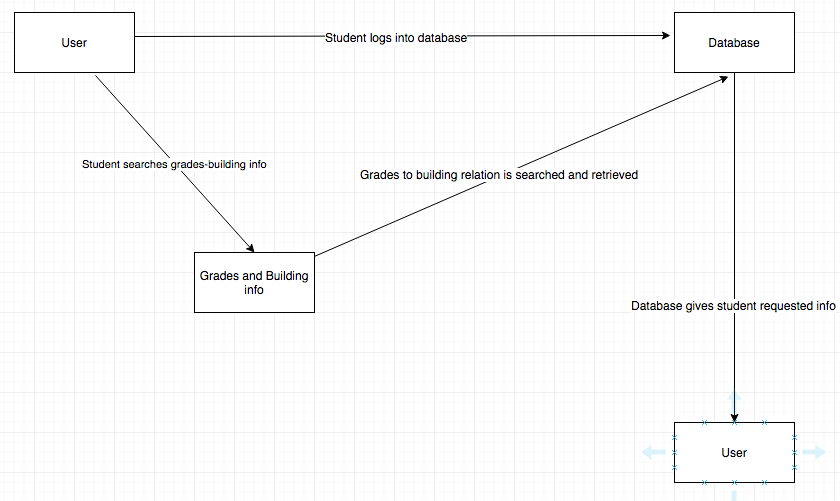


**Actors:**

* Student
* Admin

**Flow of Events:**

* Use case starts when Student logs in
* The system searches traffic information from swipe data
* Student checks traffic information
* The Database gives traffic info
* Use Case ends.

**Use Case 2:** Student checks what time visiting a building is resulting in higher grades

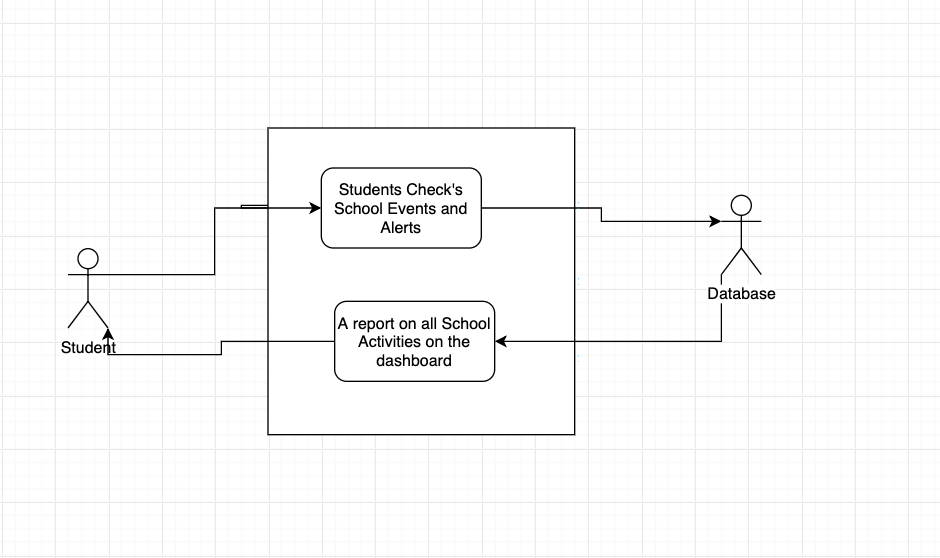
**Actors:**

* Student
* Database

**Flow of Events:**

* Use case starts when Student searches correlation between building traffic and grades
* The system searches data for traffic and grades of students and finds which building is resulting in higher grades
* Student checks report
* Use Case ends.

**Use Case 3:** Students Checks what events and alerts are happening around the school at a given time.



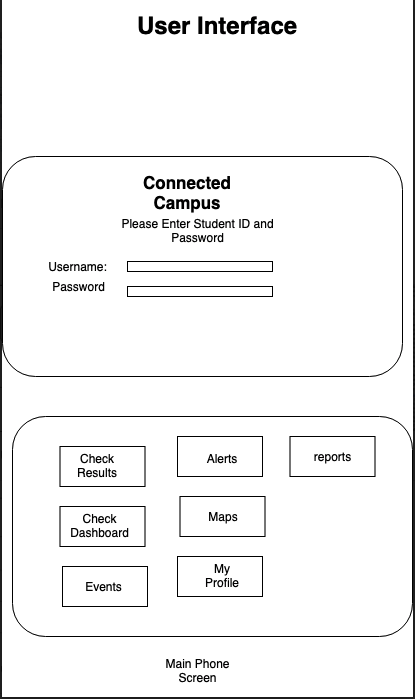
**Actors:**

* Student
* Admin

**Flow of Events:**

* Use case starts when Student swipes card.
* The system then records student record.
* Student checks school events and alerts
* The Database retrieves reports
* Use Case ends.

**Use Case 4: (Log in)**Proposed User Interface for mobile application version of Connected campus



**Actors:**

* Students
* Admin

**Flow of events:**

* Use Case Starts when Student Opens the Mobile Application
* Student then log in with ID and password
* Student then selects the different application options.
* Case use end.

**Scenarios**

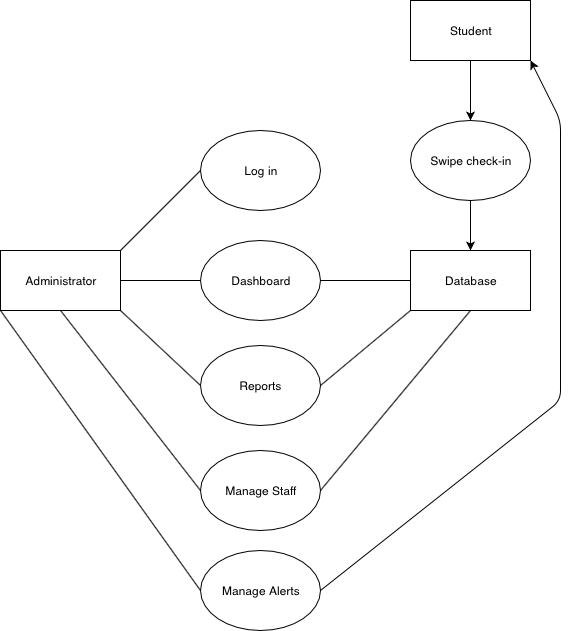
· Bad User Name

· Bad password

· Too many attempts to log in, account on hold

· Admin has the control for students logins.

**Use Case 5: (Administrator View)** As students swipe to check-in, administrators can log in and view real-time data through the dashboard and check reports on all areas of the campus. Based on the information, staff changes can be made and alerts will be sent to students accordingly.



**Actors:**

* Student
* Administrator
* Database

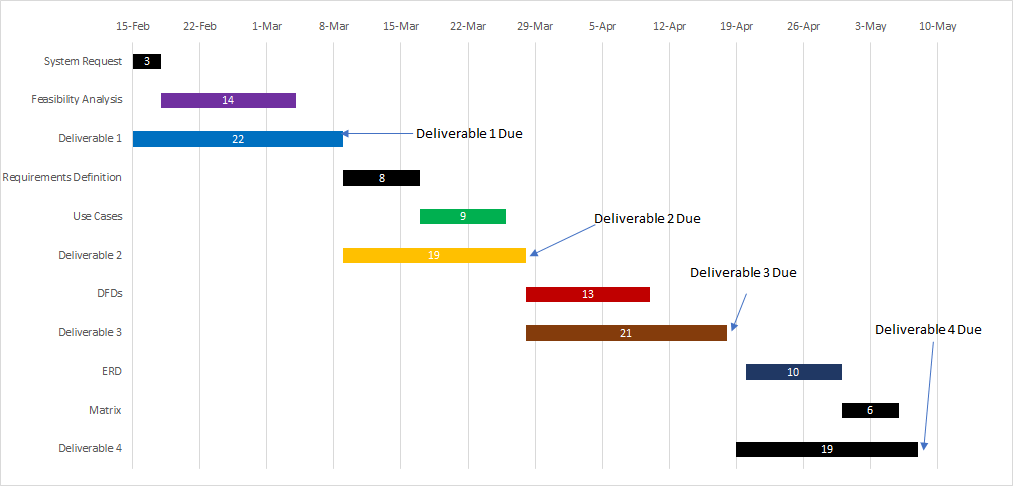
**Preconditions:**

* Database is successfully storing student data in real-time and up-to-date
* Administrator identity is authenticated

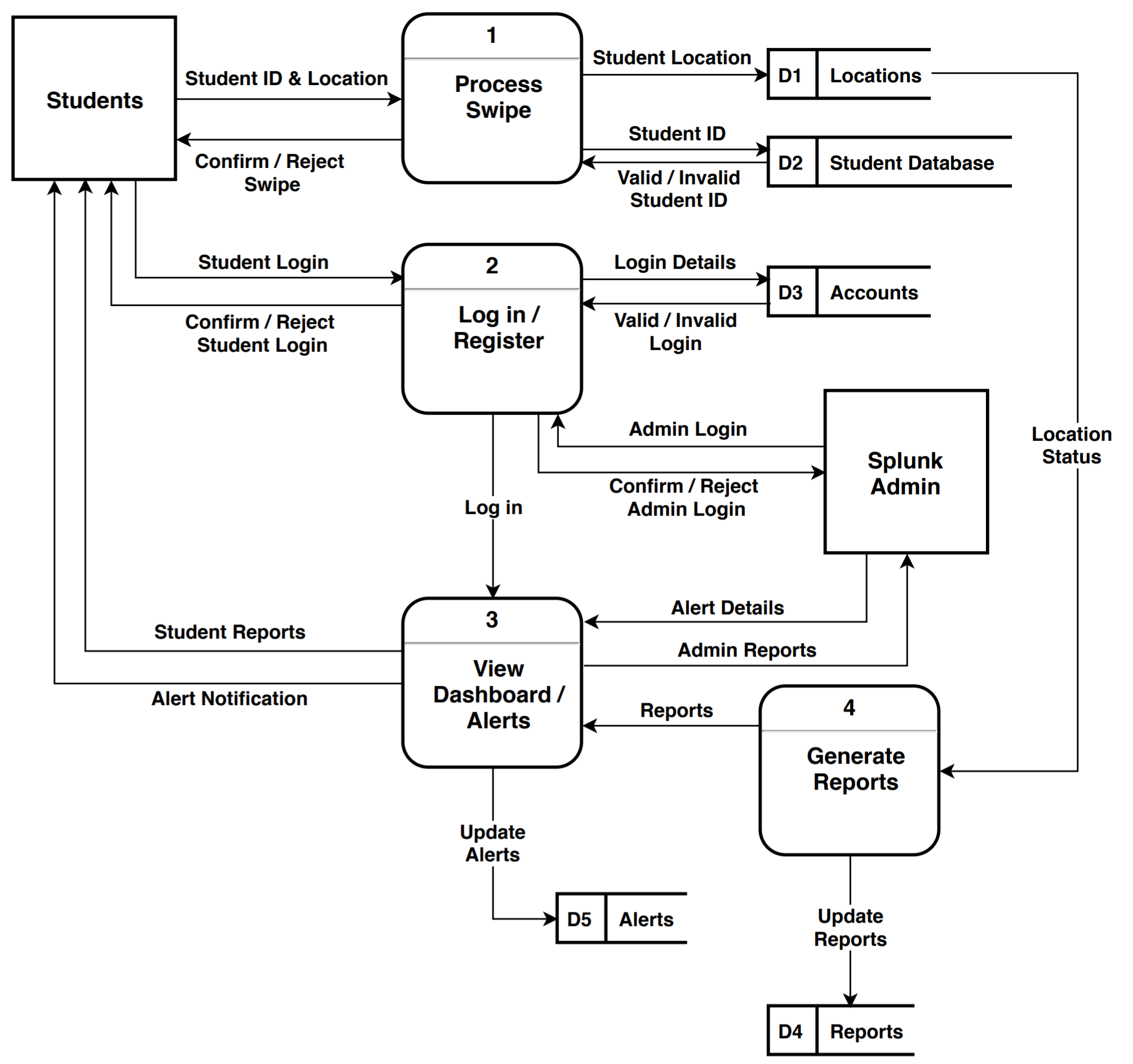
**Postconditions:**

* Dashboard retrieves data
* Reports are produced for administrator
* Staff and students alerted

**Project Plan:**



**Definitions:**

****

**Processes & Data Flows:**

* Process Swipe:
  + **Number:** 1
  + **Name:** Process Swipe
  + **Description:** This process will perform a couple of actions. First, it will validate that once a student id is swiped, they are indeed a faculty, student, or staff member of the University. If this is found to be true, the location and ID number of the individual who swiped their ID will be sent to a database where Splunk will be configured to pull data from in real-time. This would be important when analyzing traffic in areas on campus because the more swipes per specified time frame within a general vicinity, the more likely it will need to be highly staffed.
  + The **input data flow** for this process would be the student ID swipe. If this student, staff, or faculty identity is confirmed then a corresponding access control action will be initiated.
  + The **output data flows** for this process would be sending the student ID and student location information to the backend database.
* Log in/Register:
  + **Number:** 2
  + **Name:** Login/Register
  + **Description:** This process will handle the login/registering students/staff/faculty into the Splunk system to give them permissions/access to dashboards and visualizations in real-time. First, the individual will enter their login credentials into the system login screen. Those login credentials will be verified with the already existing accounts within the Splunk system. If the identity is confirmed, then access will be granted and they will be able to freely use the application within their assigned permissions/roles (if Admin role is assigned with credentials, then the corresponding dashboards/reports will be visible to the user). If the identity is unknown, the system will deny their request and ask for them to either re-enter proper credentials, or create an account using their UMBC login.
  + The **input data flows** are the login credentials entered.
  + The **output data flows** would be verifying the credentials are active in the accounts database, verifying the roles/permissions associated with the account, and displaying relevant dashboards and reports to the user.
* View Dashboard/Alerts:
  + **Number:** 3
  + **Name:** View Dashboard/Alerts
  + **Description:** Depending on the results from the previous process (login stage), certain dashboards/reports/alerts will be generated and displayed for the user. If an admin has logged in, corresponding alerts and dashboards will be available for them. If a student is logged in, then they will be able to see the dashboards they’re permitted to see, and alerts they’re able to access and receive.
  + The **input data flows** are the login credentials, and alert details from the previous process.
  + The **output data flows** include corresponding admin reports, student reports, and alert notifications. Also, if any alerts need to be updated by the user, the output will reflect the updates.
* Generate Reports:
  + **Number:** 4
  + **Name:** Generate Reports
  + **Description:** In this process, we get inputs from the location status forwarded to the indexer from the first process. These location status reports will be used to update the dashboards in real-time to reflect the most recent changes in student traffic around campus. When the reports are updated, the output will reroute back to process 3 to display these reports to the user.
  + The **input data flows** is location status.
  + The **output data flows** are updating the reports, and making sure they’re displayed in Process 3.

**Entity:**

* **Students:** The students will interact with the system using the UI on the app. They will retrieve information from the application about student traffic on campus, LMS engagement, and other statistics that they will have permissions to. The only data they will be contributing will be when they swipe their student ID’s to access different areas on campus. Without their swipes, this application will not accurately reflect the goals.
* **Splunk Admin:** The Splunk Admin’s will interact with the system on the backend, ensuring that dashboards are up to date and data flow is smooth. They will also have delete access if they would like to delete records (i.e. records indicating denied access). They could also create and edit saved searches, run searches, edit its own preferences, create and edit event types, create and edit alerts, and other similar tasks.

**Data Store:**

* **Locations:** This data store will hold all the information about the location data taken from student swipes around campus. This would make it easier for Splunk to retrieve from a specific data source, rather than parsing and indexing through a variety of data sources not specified. This will make the transaction time increasingly faster.
* **Student Database:** This data store will hold all the information about the student information (login credentials, whether current student or alum). This will be used to verify login in Process 2.
* **Accounts:** This data store will hold all the credentials about login. If the system receives a login that isn’t already existing in the database, then it will inform the user that they have entered incorrect credentials and to try again. There will also be a ‘sign up’ option where the user has the ability to create a new login. Ideally, the login credentials will be the same as their UMBC myUMBC login.
* **Alerts:** This data store will hold all the current alerts that have been set by the user to be triggered off a certain criteria. If the user would like to modify/add/delete alerts, they could do so on the UI which will send queries to the backend database to make these preferred changes.
* **Reports:** This data store will hold all the reports that will be generated upon startup, or triggered reports based on set intervals. If the user would like to modify/add/delete reports (based on their permissions as well), then they could do so.