Software Engineering

UGATE

Smart Security System

20 June 2020

Dr. Abhishek Kumar Thakur

IDRBT

Introduction:

Problem Statement:

Design a system to support a digitized security management for University including both verification software and human security officers. System is aimed to address everyone who enters the campus through the main gate.

Impact:

University of Hyderabad has around 5000 students with a faculty of 400 and a huge staff working for maintenance of the campus. Managing such numbers needs manual labour. The campus also offers maximum freedom to students, as there is no time restriction in terms of entry and exit, which makes it harder for security guards to strictly follow the rules.

The system we develop ensures safety and prevents unauthorised entries into the campus and enables students and staff to have a digitized communication with the gate, eliminating the need to reach out for an approval. System allows security officers to collect the packages dropped at the gate by delivery executives.

Team :

Member 1	Alla Leela Srija	17MCME18
Member 2	Vaishnavi Kompally	17MCME04

Budget:

Module Name	**Quantity**	**Cost Estimate**
Device to scan QR code	1	7000 INR
Automatic Boom Barriers	1	38,000 INR
Device to manage verifications.	1	5000 INR
	Total Cost	50,000 INR

Development Process:

This system requires automation and integration of different software subsystems and continuous upgrades and releases. In this case the development team is well aware of both user and system requirements. On the other hand, the operations team is sensitive about managing resources, quality, frequency and stability of releases. Hence, focusing on the need for collaboration between development and operations teams at all stages of the software development life cycle, **DevOps** is adopted.

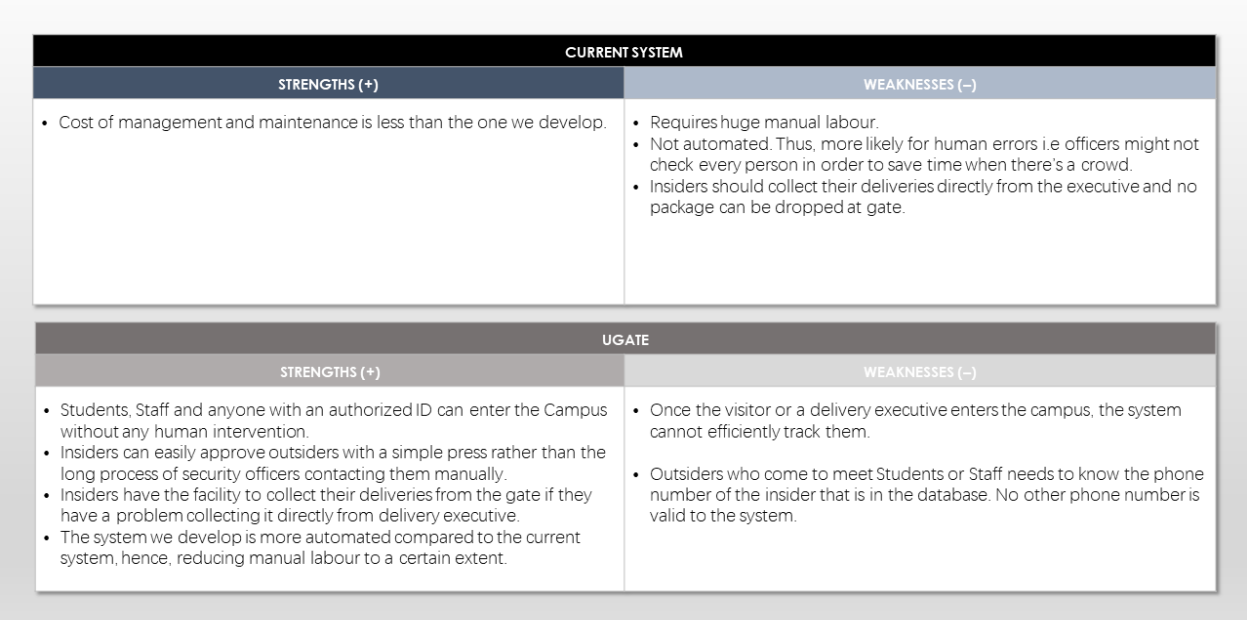
Competitive Analysis:

Features/ Competitors	Our product	MyGate	NoBrokerHood Visitor Gate & Security Management	JioGate
Strengths	1. Can manage large numbers efficiently 2. Suitable for universities and colleges 3. Automatic insider verification	1. Suitable for small residential communities 2. Community management	1. Suitable for small residential communities 2. Community management	1. Suitable for small residential communities 2. Community management
Weaknesses	Visitor and delivery executive tracking is not implemented	Unsuitable for universities and institutions with large population	Unsuitable for universities and institutions with large population	Unsuitable for universities and institutions with large population
Target Customers	Universities, colleges and large companies	Gated communities and apartments	Gated communities and apartments	Gated communities and apartments
Onboarding experience	Easy user interface	Smooth instructions	Easy user interface	Enables anyone to create a society and use it with simple UI

Gap Analysis:

This analysis benefits in following ways

- Insight into areas that need improvement.
- Ensuring that project requirements have been met.



CURRENT SYSTEM	
STRENGTHS (+)	WEAKNESSES (–)
• Cost of management and maintenance is less than the one we develop.	• Requires huge manual labour. • Not automated. Thus, more likely for human errors i.e officers might not check every person in order to save time when there's a crowd. • Insiders should collect their deliveries directly from the executive and no package can be dropped at gate.

UGATE	
STRENGTHS (+)	WEAKNESSES (–)
• Students, Staff and anyone with an authorized ID can enter the Campus without any human intervention. • Insiders can easily approve outsiders with a simple press rather than the long process of security officers contacting them manually. • Insiders have the facility to collect their deliveries from the gate if they have a problem collecting it directly from delivery executive. • The system we develop is more automated compared to the current system, hence, reducing manual labour to a certain extent.	• Once the visitor or a delivery executive enters the campus, the system cannot efficiently track them. • Outsiders who come to meet Students or Staff needs to know the phone number of the insider that is in the database. No other phone number is valid to the system.

Stakeholders:

- Students
- Faculty
- Daily staff working for the university
- Administration
- Visitors
- Delivery Executives
- Transportation services within university
- Cabs

Requirement Analysis:

I. Functional Requirements

User Requirements:

1. Every person with an authorised ID shall be able to enter the campus.
2. An insider i.e. students,faculty or staff shall be able to easily approve their visitors with digitized communication with gate.
3. A visitor, delivery executive or a cab driver shall be able to enter the campus after verification of insider by system and approval from the insider.
4. A delivery executive can leave the packages/deliveries at the gate if instructed by their customer belonging to the university.
5. One should be able to collect packages from the security, within the given time. In case of food deliveries, packages are expected to be collected within 1 hour from the time of drop. Other deliveries are supposed to be collected within 6 pm of that particular day.
6. People entering the university for admissions, examinations and other special events should be directed to the security officer.

System Requirements:

1. The system shall be able to verify the authorized people by scanning their ID.
2. The system shall be able to prevent unauthorized entries to the campus.
3. The system shall be able to verify visitors with an approval from insiders.
4. The security officer should keep the delivery executive's ID and send him in.
5. The security should be able to collect packages from delivery executives and secure them until they are picked within the given time.

II. Non-Functional Requirements

Organizational requirement:

- Users are not allowed to add any other user or change certain information.

Performance requirement:

- The system shall work for 24 hours on all days unless the administration orders otherwise.

Security Requirement:

- Information of the insider should not be disclosed to anybody.
- Information of visitors and delivery executives is not stored.

Capacity requirement:

- System should be capable enough to handle multiple users at the same time.

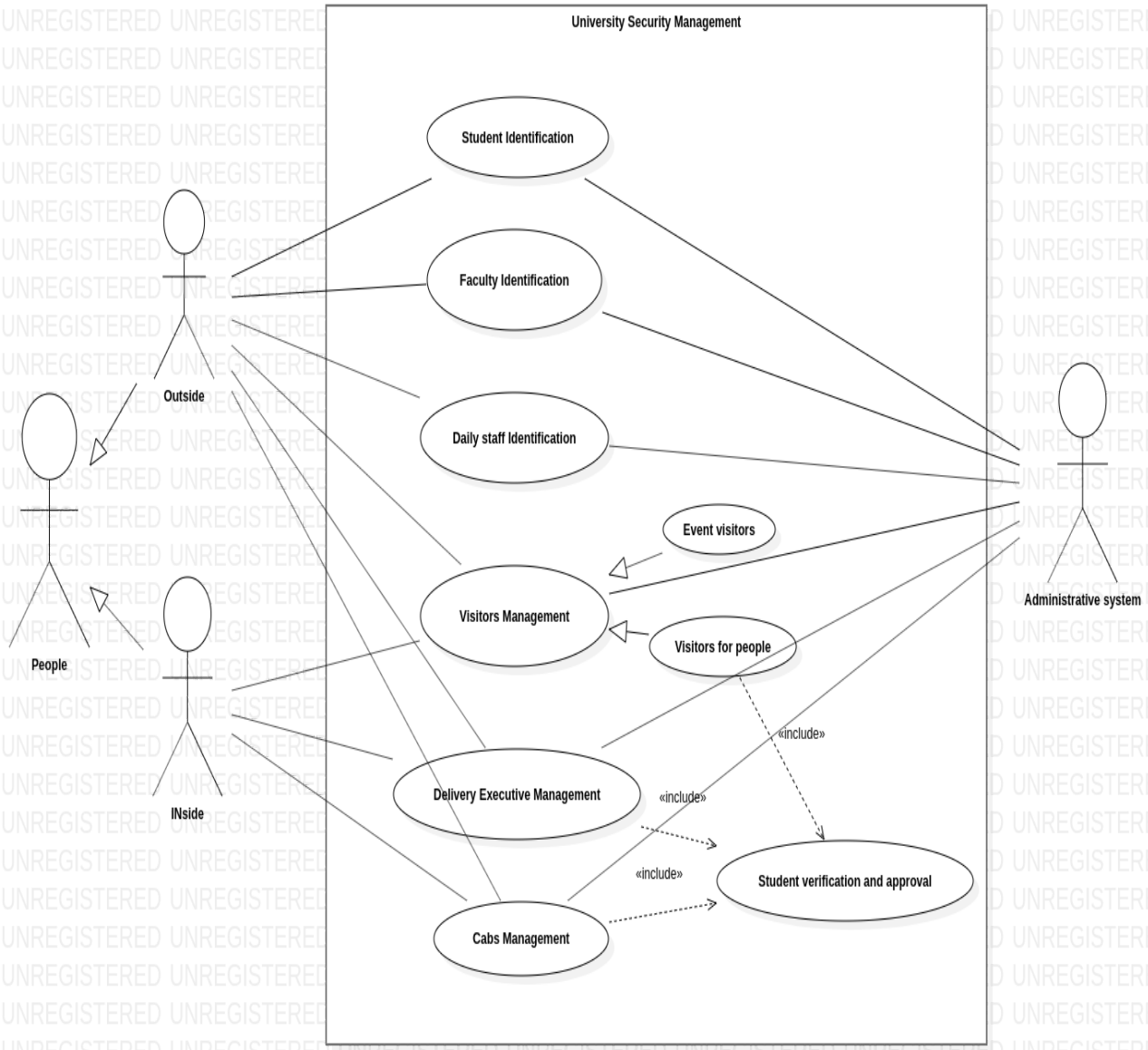
Dependability requirement:

- Software Failure: If the insider does not receive any notification for approval of an outsider, security manually manages the issue with a phone call in most cases.
- Hardware failure: If the gate fails to open after insider verification security would open it manually.
- Operational failure: If a person fails to collect their package from the gate in the given time their packages are sent to storage and no security guard at the main gate is responsible. Only the managers of the storage are responsible for 5 days.

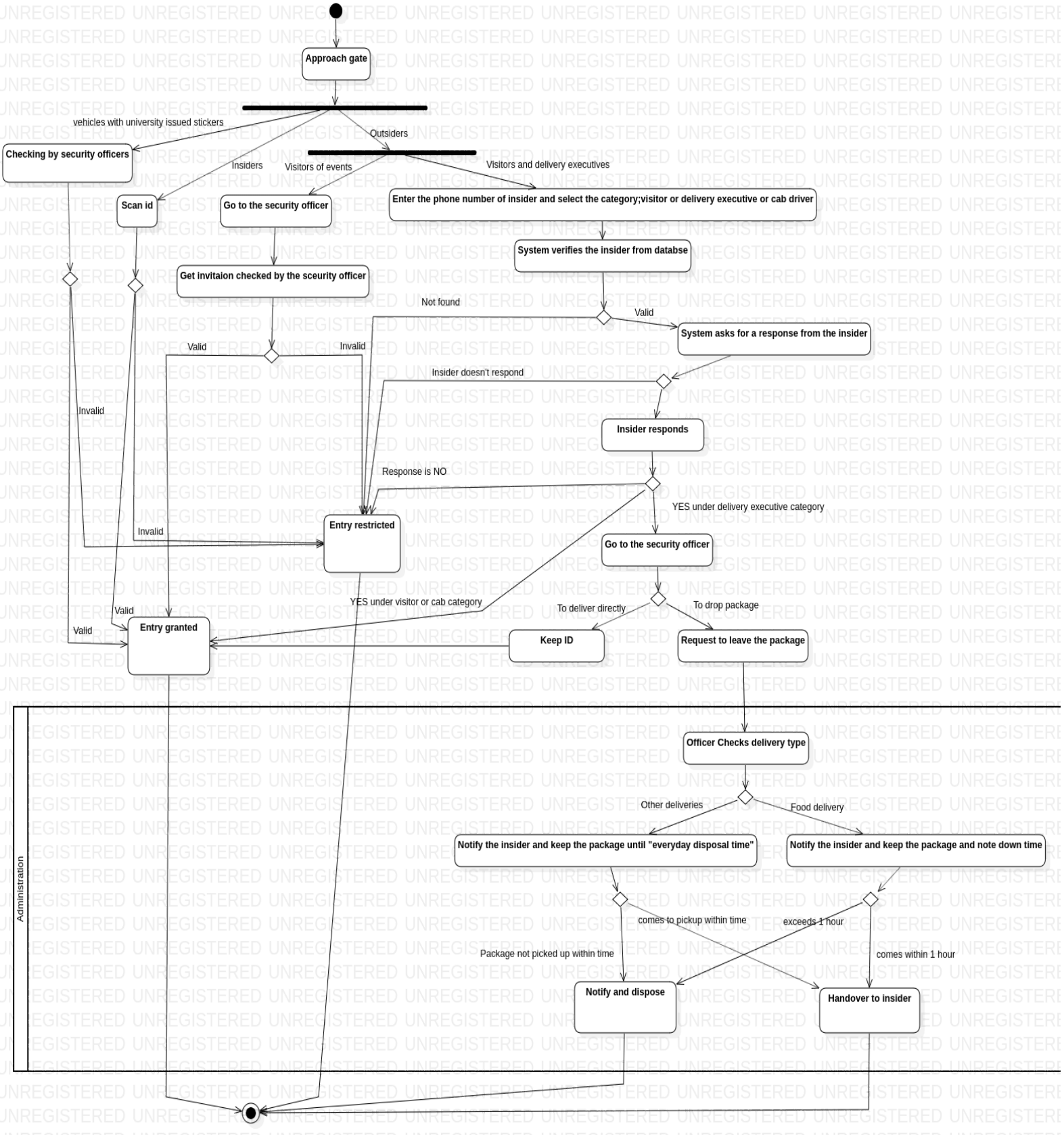
Usability requirement:

- The user interface should be simple, consistent and easy to use for students, Faculty and outsiders as well.
- To reduce the time taken for verification and make it more automated, outsiders are asked for minimal information which in this case is a phone number.

1. *Use Case Diagram*

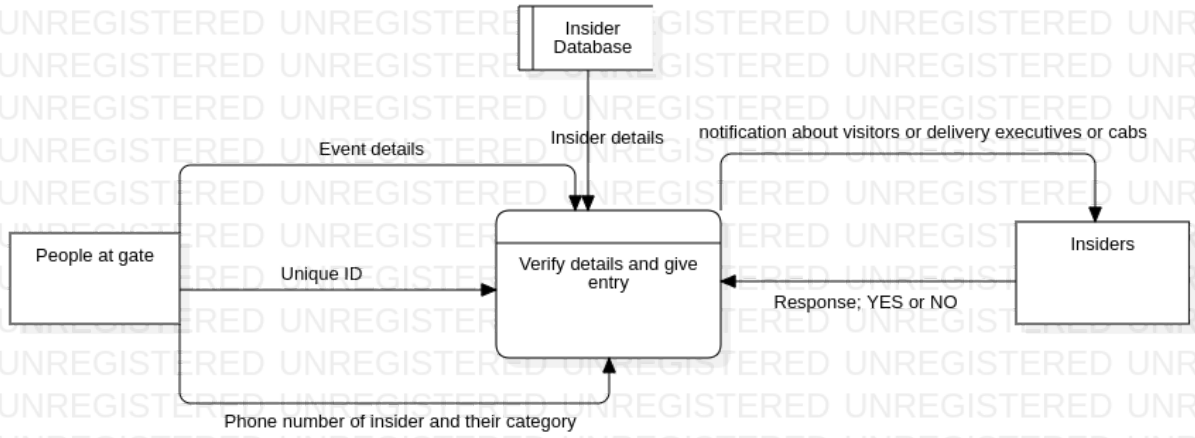


2. *Activity Diagram*

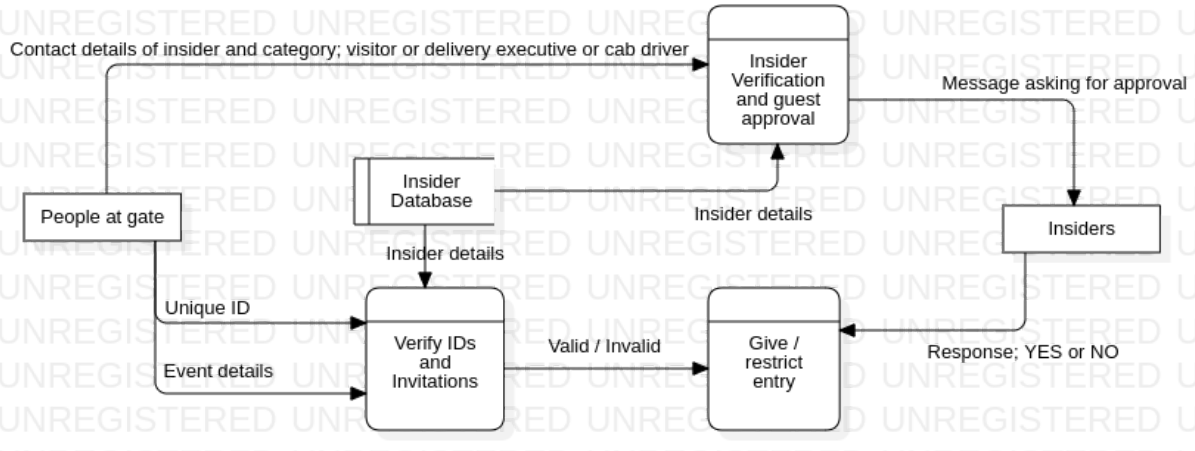


3. Data Flow Diagrams

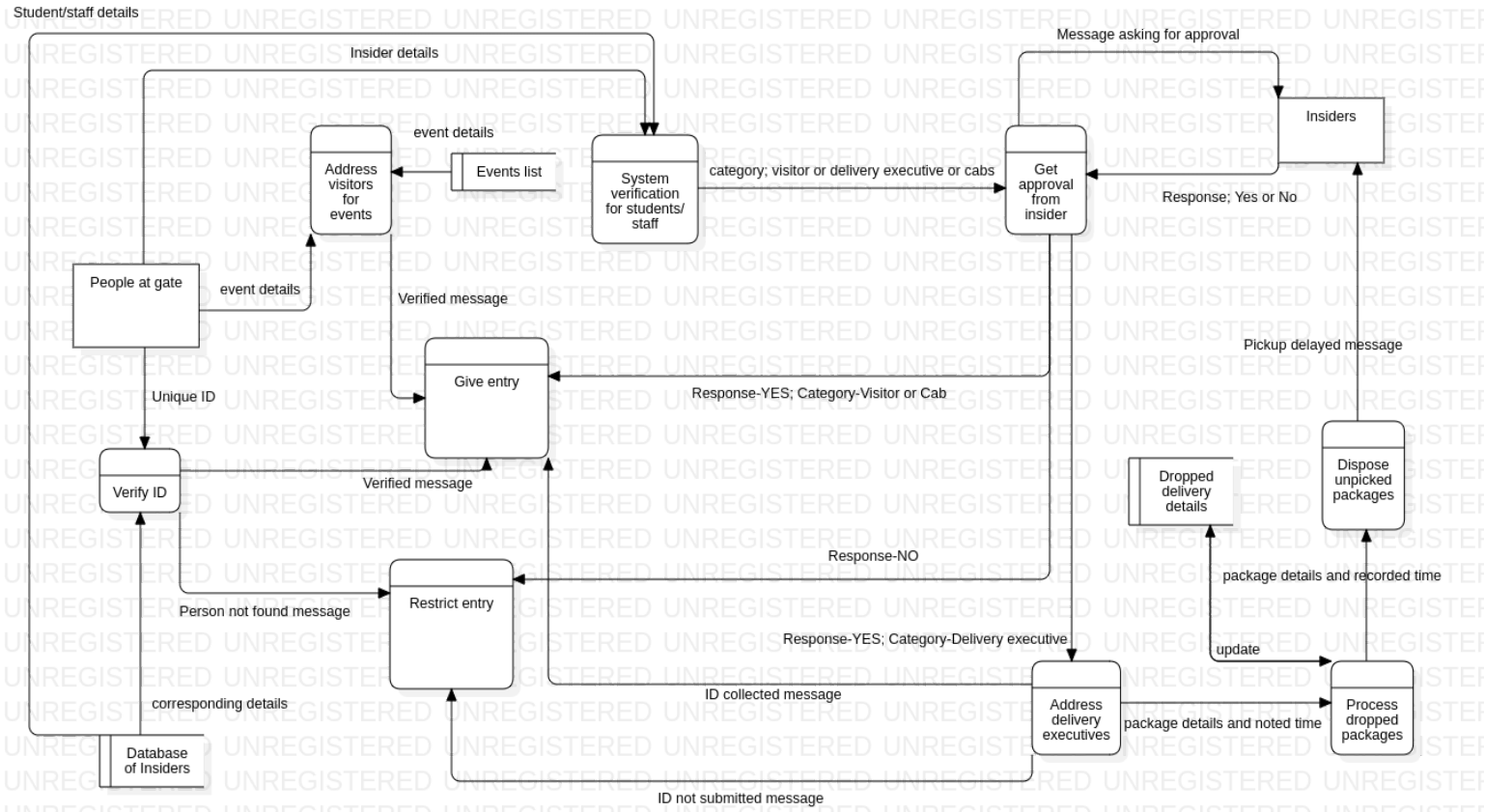
a) Level 0 DFD



b) Level 1 DFD



c) Level 2 DFD



Alternative Architecture:

Agile Methodology can be an alternative approach to this project. It emphasizes on iterative, incremental and evolutionary development.

Part of the effort in agile architecture is for initial requirement envisioning and architecture envisioning so that critical questions about the scope, cost, schedule and technical strategy of the project can be answered.

It breaks down the process into smaller units and integrates them for final testing. The end result of the architecture emerges in increments faster at first because of the need to set a foundation to the project.

It provides the ability to rapidly adapt to the changing requirements and reduces technical risk of the project.

Tool used if Agile architecture is used:

JIRA supports any agile methodology such as scrum, kanban etc. It is mainly an issue tracking tool that assists agile teams to plan and manage their projects.

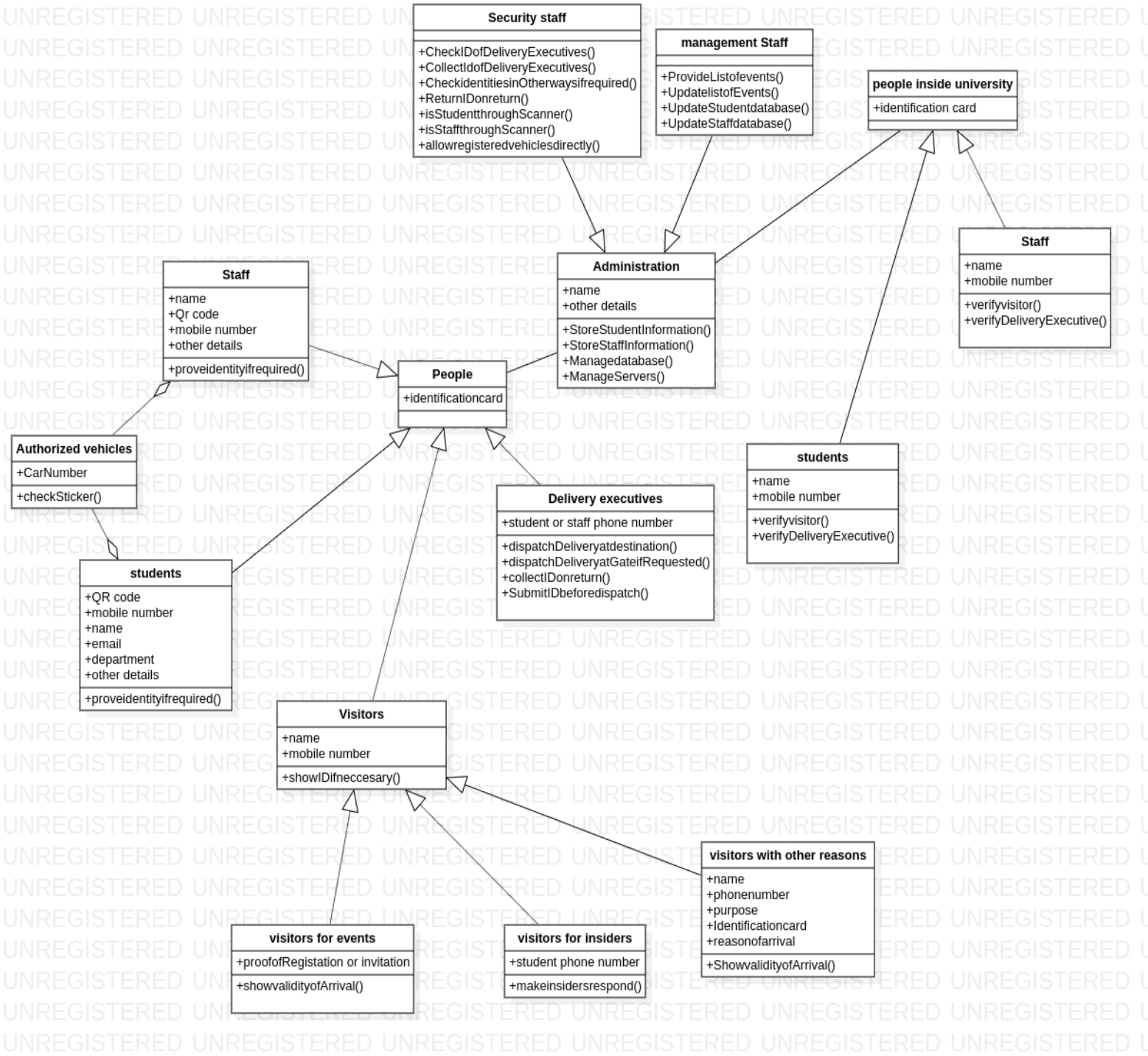
Detailed Subsystems:

Subsystem is a group of interconnected and interactive parts that performs an important job or task as a component of a larger system.

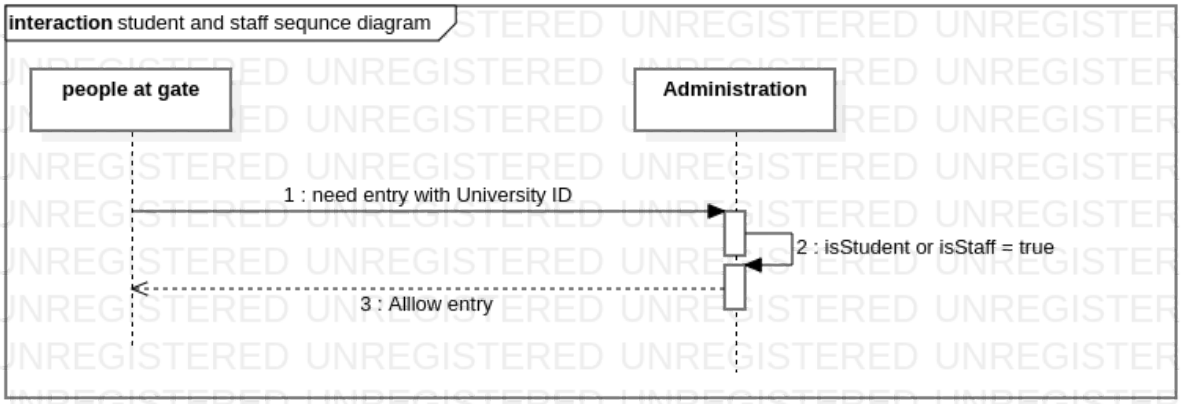
Subsystems in this architecture:

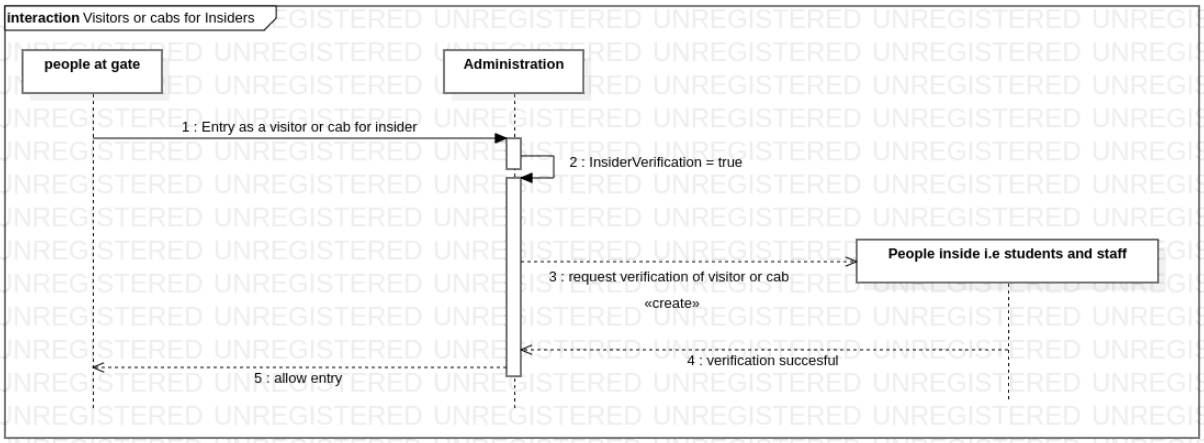
- Scanning and Verification of authorized IDs.
- Verification of Authorized Vehicles with university issued stickers.
- Verification of insider information entered by an outsider.
- Insider approving their visitors, delivery executives and cabs.
- Security officers collect delivery executives' ID and give back on their return.
- Management of deliveries dropped at the gate.
- Management of visitors for events, examinations and others.

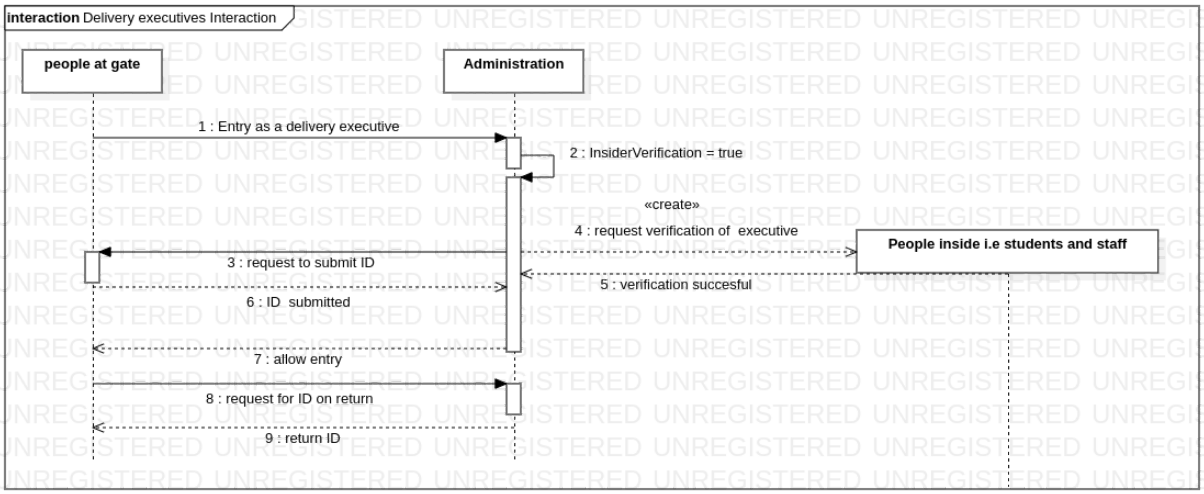
4. Class Diagram

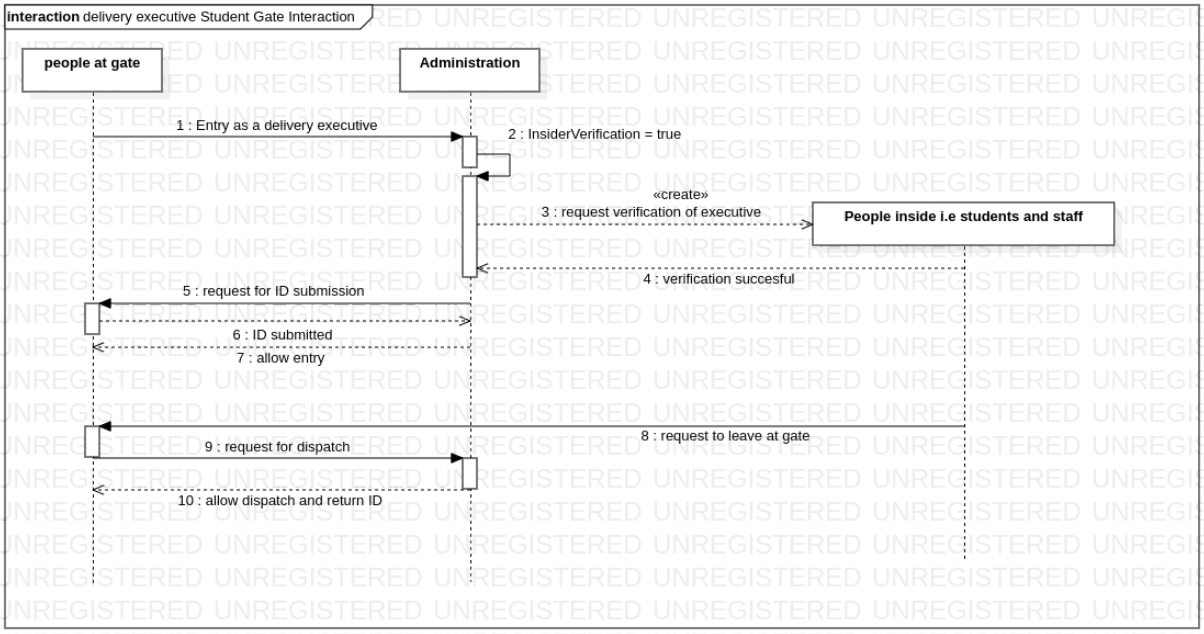


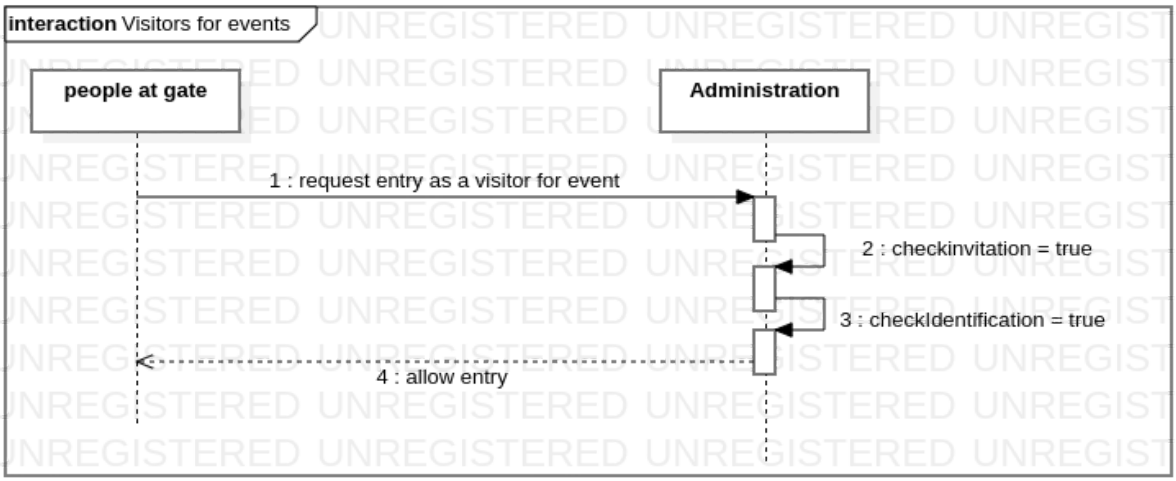
5. Sequence Diagrams

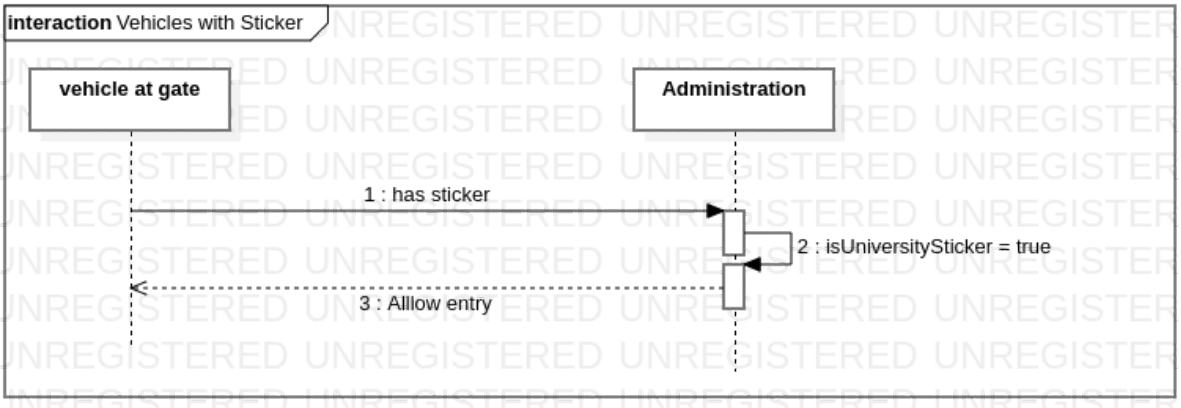




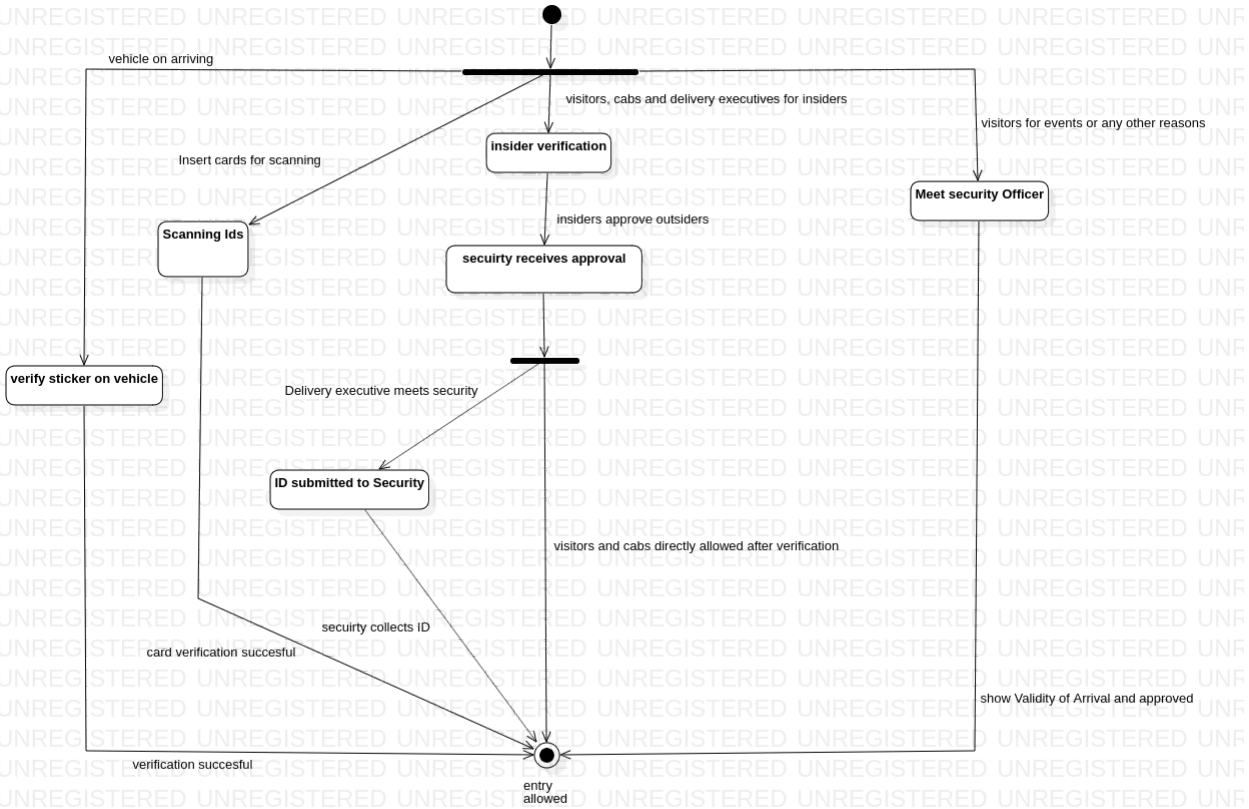



interaction delivery executive Student Gate Interaction
people at gate
Administration
1 : Entry as a delivery executive
2 : InsiderVerification = true
«create»
3 : request verification of executive
People inside i.e students and staff
4 : verification succesful
5 : request for ID submission
6 : ID submitted
7 : allow entry
8 : request to leave at gate
9 : request for dispatch
10 : allow dispatch and return ID


interaction Visitors for events
people at gate
Administration
1 : request entry as a visitor for event
2 : checkinvitation = true
3 : checkIdentification = true
4 : allow entry

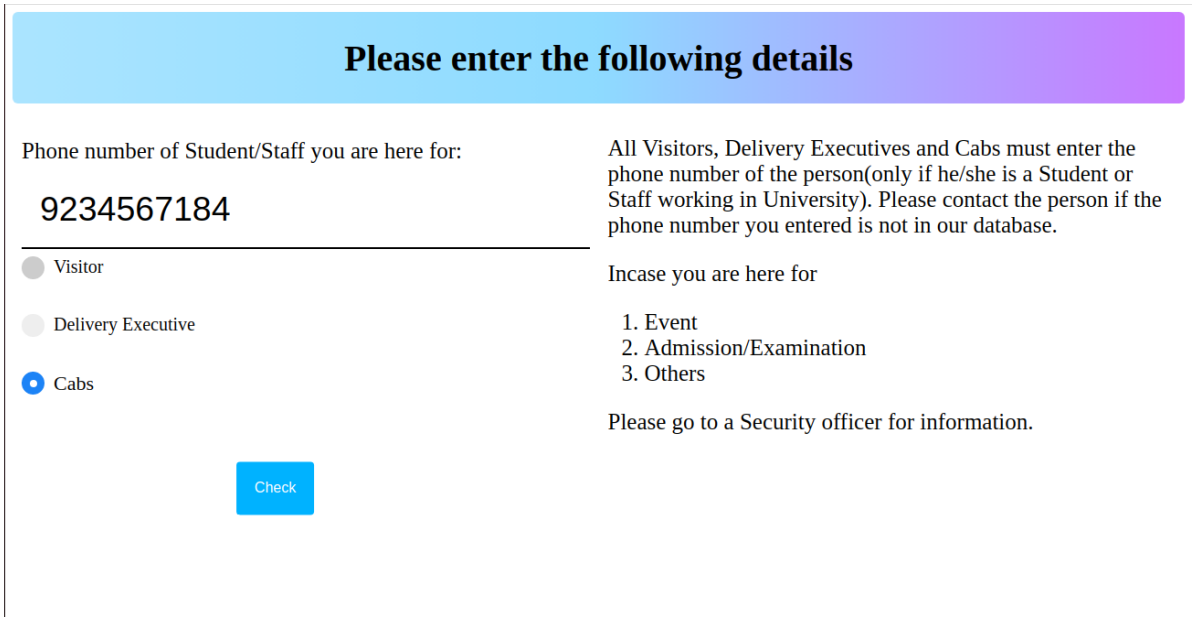

interaction Vehicles with Sticker
vehicle at gate
Administration
1 : has sticker
2 : isUniversitySticker = true
3 : Alllow entry

6. State Diagram



Mock User Interface:

Scan this QR code to see the Mock application.



Test Plan:

Objectives:

- Identify the software that should be tested.
- Identify the required resources.
- Test the interfaces between different subsystems.
- Test the performance measures of different functions.
- Describe the testing strategies and tools.

Features to be tested:

- Authorized ID verification by scanning.
- Automatic verification of insiders.
- Insiders approving visitors, delivery executives and cabs through application.
- Security officers collecting IDs of executives.
- Security officers collecting packages at the gate.
- Proper disposal of uncollected packages within the proposed time.

Testing types:

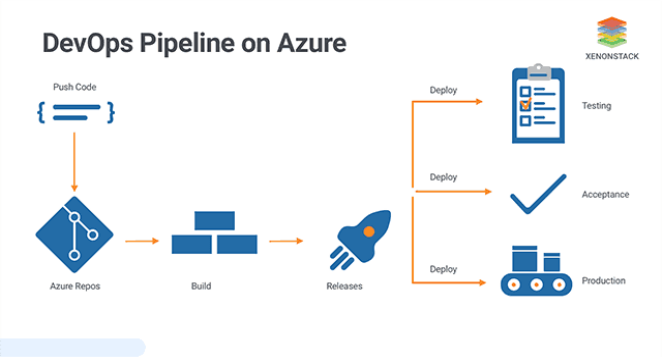
- Unit Testing
    - Test components individually such as verification of authorized IDs by scanning, insider verification by guests and approving visitors and executives.
- Defect Testing
    - Identify the defective QR codes or the ones not issued by the university.
    - Check for any missing information of students or faculty in the database.
    - Ensure that there are no server problems.
- Usability Testing
    - Ensure that the system is well documented and user-friendly.
    - System should be able to navigate users if they are stuck at any point.
- Back-end Testing
    - Ensure Database access methods and processes function properly and without data corruption.
- Regression Testing
    - Test the system as a whole for modifications in any functionality.
- Recovery Testing
    - Test whether the system continues to operate after disasters such as hardware and software failures.
    - Security officers must be aware of all the protocols.
    - Security officers are supposed to take situations under control in most cases.
- Beta Testing
    - Ensure that there are no major failures in the software or product and it satisfies the business requirements from end-user perspective
    - Testing a beta-version before the product is released to end-users.

Tools and Automation:

Framework: Azure Devops Framework

Code and Build: Git integrated with Azure Pipeline

Testing and Deployment: Azure Pipeline



Continuous Testing:

The goal of Continuous Testing is to test early and test often. The process involves stakeholders like Developer, DevOps, QA and Operational system. There is no much need for Test Automation in this case as it is not a model that merely involves a bunch of codes, but there is a need for continuous testing.

Pair testing:

The testing of the software system is done by two team members at a single workstation. One does the testing and the other reviews and analyzes the test results. It breaks down the communication barrier.

Change Management:

As we plan on following DevOps culture, the model combines continuous integration, continuous testing and continuous delivery allowing the team to react in a timely manner.

By including change management in CI/CD pipelines, the team can reduce the risks associated with changes, while getting all DevOps benefits like reduced deployment time, transparency and traceability.

Change management frameworks are designed to deal with large-scale changes, identifying dependencies and operational risks. DevOps reverses this approach by optimizing for small frequent changes,breaking big changes into smaller iterative steps and trying to automate how they are managed.

For instance, if the customer requests for delivery executive tracking, the system needs their information, in-time and information of the insider they are delivering to. It also needs to alert security when the permitted time exceeds.

In this case steps involved:

- Announcing protocols for Delivery Executives.
- Taking executive's information when they request for insider approval.
- Collect their ID and record the time stamp when the insider approves.
- Stop recording the time, return ID and delete their entry in records when they leave.
- Alert security if they do not leave within permitted time.

In the implementation of the above transition, once a milestone is achieved, do a retrospective to assess the current state. If it's in-line with the expected results, initiate the next step.

After the transition, perform testing and identify the issues that have come up as a result of the new process.

Change Management Tool:

Integrated with Azure DevOps:

Specific change management sub processes in this tool include change risk assessment, change scheduling, change approvals and oversight.

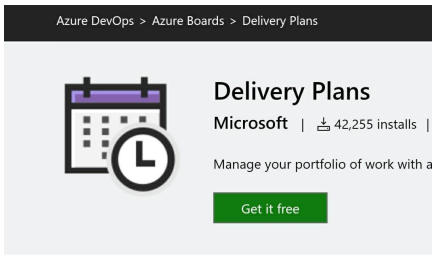
This change management extension for Azure DevOps helps automate the process involving, reduce chances of human error and increase deployment agility.

servicenow

Project Plan:

Tool for Planning:

Delivery Plans is an extension of Azure DevOps and is easy to use



Schedule:

Milestone Task	Start Date	End Date
Planning	March 3rd	March 23rd
Design	April 1st	April 8th
Development	April 10th	In process
Coding a subsystem	April 12th	April 14th
Building a subsystem	April 15th	April 17th
Testing a subsystem	April 18th	April 21st

As we follow DevOps approach, after developing the first subsystem we start developing and integrating all other subsystems. They go through several iterations of development and testing.

These iteration paths define the cadence of delivery for a team.

Azure Pipeline tests them continuously and delivers them to our choice of end point.

Risks:

- Human errors:
    - Students can enter the campus by approving themselves as visitors when they forget their IDs.
    - Delivery executives can enter as visitors if approved by their customers.
    - Wrong entries by visitors or delivery executives or cab drivers.
- Hardware Failures:
    - In cases where the scanner fails to recognize a valid code.
    - Gate mechanism failure.
- Software failure:
    - When there are server problems and the student can't receive notification but the security is unaware of the issue.

Risk-mitigation:

- To prevent scanner and gate issues, they should be checked and monitored frequently.
- Employers should constantly check the status of servers on both sides.

Conclusions:

Learnings & Challenges:

- The whole security system cannot be automated. There are certain issues in security which only humans can solve.

- Human errors cannot be totally exhausted.
- Integrating the system with e-governance.
- High budget and maintenance issues.
- Executional challenge: COVID-19.

Possible Extensions:

- This security system can be implemented at all gates in the University and integrate all gates.
- Monitor entries and exits of visitors, delivery executives and cabs.
- Overstay alert for delivery executives.