Q1. These are the flowchart steps. You will have to draw the flowchart properly, using all the right shapes.

1. Start
2. Read occupancy %, day, event status, weather
3. Adjust threshold based on weekend/event/rain
4. If occupancy ≥ threshold → open overflow parking
5. Else → keep closed
6. End

Note: These are brief, broader steps. You can further improve this by adding more decision boxes as per requirement.

Q2. These are the flowchart steps. You will have to draw the flowchart properly, using all the right shapes.

1. Start
2. Read soil moisture, time, crop type, forecast
3. If rain expected and soil not critical → skip watering
4. Check crop moisture requirement
5. If moisture below target and time is suitable → water
6. Log action
7. End

Q3.

Given:

Alert triggers when:
1. Door open AND (motion OR alarm armed)
OR
2. Door closed AND alarm armed AND nighttime

Let variables: D=1 door open, M=1 motion, A=1 armed, N=1 night.

<u>Boolean expression</u>

$S=D(M+A)+D'AN$

Then, simplify it :

$S=DM+DA+D'AN$

$DA+D'AN=A(D+D'N)$

$D+D'N=(D+D')(D+N)=1(D+N)=D+N$

So, $S=DM+A(D+N)$

Q4.

$M = TV + T_mA$ – I have assumed as follows:

T = Temperature high

V = Vibration detected

T_m = Temperature moderate

A = Machine age > 5 years

Q5.

Part 1 - What goes into SQL vs NoSQL

SQL (structured, relational, consistency-critical):

- Patient master records (MRN, name, DOB, allergies)
- Appointments/admissions/discharges
- Clinician notes metadata, diagnoses codes
- Insurance/billing

NoSQL (high-volume, flexible, semi/unstructured):

- Real-time heart-rate time-series (many readings per patient per day)
- Mobile app feedback (free text, ratings, optional fields)
- Device telemetry/logs

Part 2 - Two SQL table examples

Table 1: Patients

- patient_id (PK)
- name

- dob
- gender
- phone
- blood_group
- created_at

Table 2: Admissions

- admission_id (PK)
- patient_id (FK → Patients.patient_id)
- ward
- bed_no
- admit_time
- discharge_time
- primary_diagnosis_code

Part 3 – You can mention any suitable NoSQL type here for example MongoDB is more suitable here.

Q6.

Data types:

- Student profiles (structured)
- Course enrollments (relational)
- Clickstream activity (high-volume events)

Hybrid solution

- **SQL DB (PostgreSQL/MySQL):**
    - Students, Courses, Enrollments, Grades
- **NoSQL/event store (MongoDB/Cassandra + storage):**
    - Clickstream events: page views, video plays, quiz attempts, timestamps

Data flow

1. User logs in leading to profile/enrollment being fetched from SQL
2. Every click/view generates an event → written to event collector
3. Events stored in NoSQL
4. Nightly/near-real-time pipeline aggregates events → creates:
    - Daily engagement summaries (per student/course)
    - Dropout risk features
5. Aggregates written back to SQL (for dashboards)
6. Admin dashboard reads:
    - Official records from SQL
    - Engagement analytics from aggregates/warehouse

Q7.

Git command flow:

Just keep this in mind: Clone → Branch → Add → Commit → Push → Merge

Get the project : This will copy the project to your computer.

```
git clone <repo-url>
```

Then, create your own work branch:

```
git checkout -b feature-name
```

Then, save your work:

```
git add .

git commit -m "Work done"
```

Then, upload your work – basically, send your work to GitHub.

```
git push origin feature-name
```

Now, combine work (merge)

```
git checkout develop

git merge feature-name
```

If something goes wrong (conflict case)

```
git pull

git add .

git commit -m "Resolved conflict"
```

<u>That’s it. Nothing else is strictly required for now.</u>

Q8.

Again, we will only use these 5 commands:

- git clone
- git checkout -b
- git add .
- git commit -m "message"
- git merge

Use these and follow the steps in Q7. Just keep this in mind:
Developers work on feature branches and merge changes using git merge. Disaster reports are saved locally when offline and synced to the server when internet is available, with conflicts resolved using latest updates or admin review.

Q9.

1. For spoofing, use device authentication
2. For eavesdropping, use encryption
3. For DoS, use rate limiting

These are my 2 cents. You can think of other cybersecurity attacks that may take place here and provide a safeguard accordingly.

Q10.

For interception, use TLS
For unauthorized access, use encryption
For downtime, use backups.
Again, you can think of other cybersecurity attacks that may take place here and provide a safeguard accordingly.

Q11.
3 daily metrics

- Total unique visitors
- Conversion rate (visitors → buyers)
- Average dwell time (time spent in zones)

2 visualizations

- Line Chart to track hourly unique visitors & conversions
- A heatmap to see traffic density

1 predictive insight

- Predict next-day sales based on historical patterns

Q12.

Metrics (examples – you can add more)

- On-time performance
- Average delay (minutes) by route

Visualizations

- Line chart to see on-time trend over day/week

Predictive outputs

- Predict routes likely to be delayed in next 1–2 hours
- Predict maintenance risk based on breakdown history and mileage

Q13.

3 input features

- Historical consumption (last hour/day/week)
- Temperature / humidity
- Calendar features such as hour-of-day, day-of-week, holidays/exams

ML Algorithm – Regression, because electricity demand is a continuous numerical value, not a category.

Q14.

Features

- Days since last login
- % course content completed
- Assignment submission rate / missed deadlines
- Quiz attempts, average score trend
- Forum/activity participation
- Clickstream

Dropout prediction involves assigning students to categories such as “at risk” or “not at risk”, so classification is appropriate.

Q15.

This is the expected data flow which is to be represented in diagram (flow chart) form:

Sensors → Gateway → Cloud → DB → Alerts

You can further expand it like this: (Just make sure to show it in neat and labelled boxes)

1. Sensors continuous readings
2. We are looking for this here: if thresholds exceeded → “Suspected_Fire”
3. Cloud rules engine:
   - validates device identity
   - triggers incident
4. Data stored:
   - readings - NoSQL
   - incident ticket - SQL
5. Notifications sent to:
   - SMS via security staff app
   - building management dashboard
6. You can further add trigger sprinkler/alarms via secured control channel

Q16.

This is the expected data flow which is to be represented in diagram (flow chart) form:

Sensors → Validation → Cloud → Analytics → Dashboard

Now you can further expand it as done in Q15.