# **QUESTION 1**

Q1.1 Need for the Project and Problem Identification

The need for the SafeCommunity project was identified through community feedback and crime statistics, highlighting the need for a user-friendly platform to report suspicious activities and collaborate with local authorities.

The project addresses the following problems:

1. Inefficient reporting of suspicious activities.
2. Lack of community engagement in crime prevention.
3. Limited access to emergency services.
4. Limited access to emergency services.

By developing the SafeCommunity mobile app, these problems will be addressed, enhancing community safety and facilitating collaboration between citizens and law enforcement.

Q1.2 Technological Trend - Cross-Platform Mobile Development

**What is the technological trend?**

Cross-platform mobile development using frameworks like React Native or Flutter.

**Benefits to the project timeline:**  
Faster development and deployment, reduced costs, and increased efficiency.

**Three potential risks:**

1. Performance issues due to compatibility problems.
2. Limited access to native device features.
3. Increased complexity in debugging.

**Benefits to users:**  
Seamless experience across Android and iOS devices, easy updates, and enhanced security features.

Using cross-platform development benefits users by providing:

1. Consistent user experience across devices.
2. Faster updates and bug fixes.
3. Enhanced security features.

Q1.3 Project Charter Deliverables

Here are five major deliverables that would form part of the SafeCommunity project charter:

1. **Functional Mobile Application**: A user-friendly mobile app for reporting suspicious activities, sharing emergency information, and collaborating with local authorities.
2. **Backend API and Database**: A secure backend API and database to store and manage user reports, emergency information, and community data.
3. **Integration with Emergency Services**: Successful integration with local emergency services, enabling efficient response to reported incidents.
4. **User Training and Documentation**: Comprehensive user documentation and training materials to ensure seamless adoption.
5. **Testing and Quality Assurance**: Thorough testing and quality assurance to ensure the app's stability, security, and performance.

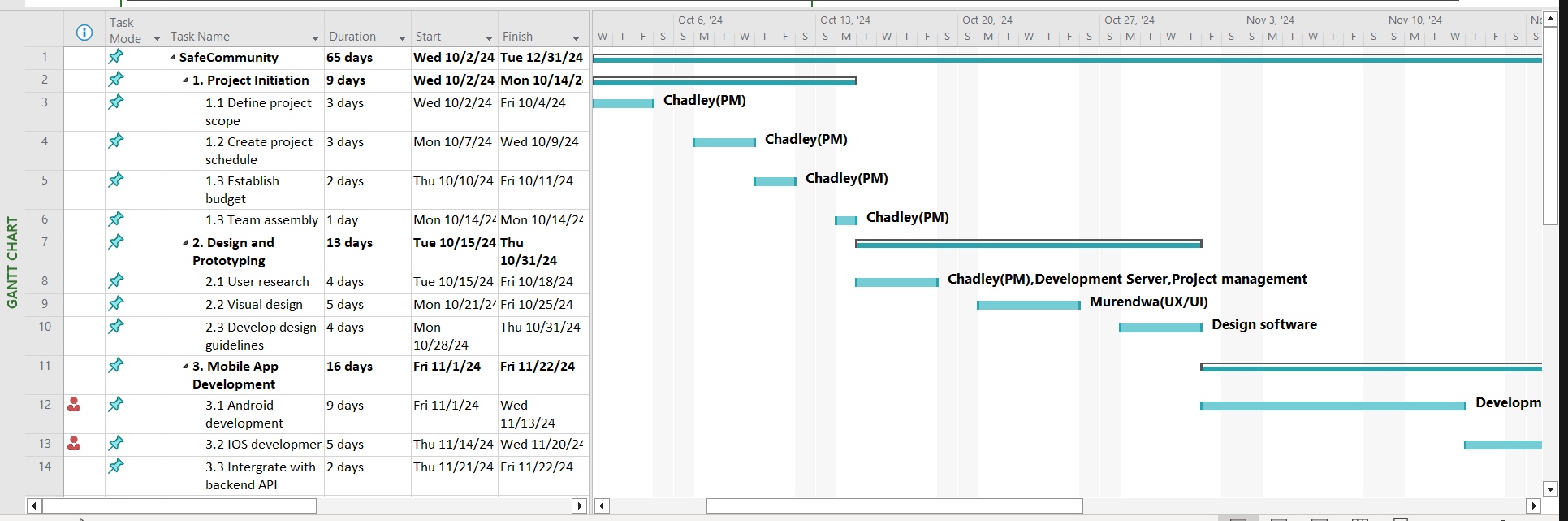
These deliverables ensure the project meets its objectives, providing a functional and user-friendly platform for enhancing community safety.

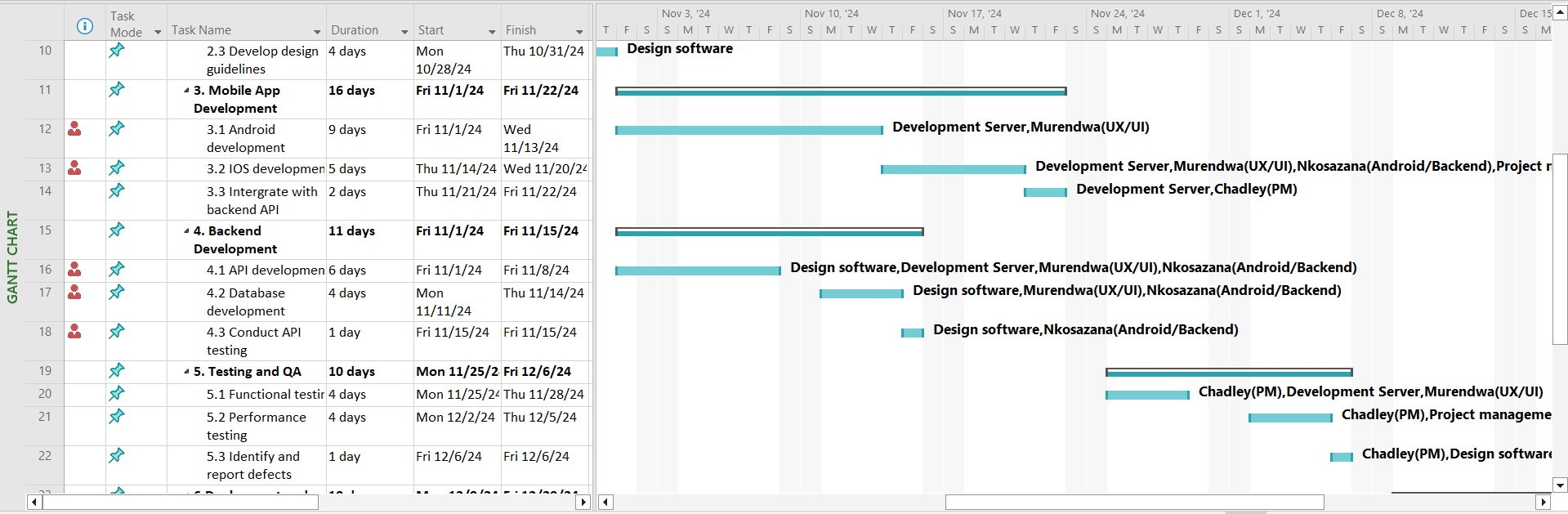
# **QUESTION 2**

# **A screenshot of a project plan Description automatically generated**

# **QUESTION 3**

## **Q3.1 Work Breakdown Structure (WBS)**





A screenshot of a computer

Description automatically generated

## **Q3.2 Issues Log**

### Issues Names:

### 

### Assignees:

### 

### Labels:

### A screenshot of a computer Description automatically generated

### Projects:

### A screenshot of a computer Description automatically generated

### Milestones:

### A screenshot of a computer Description automatically generated

# **QUESTION 4**

Q4.1 **Network Diagram with Earliest and Latest Start/Finish Times and Critical Path**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A close-up of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

A pink rectangle with black text

Description automatically generated

A close-up of a computer screen

Description automatically generated

A computer screen shot of a computer

Description automatically generated

**Table 1: Earliest Start (ES) and Earliest Finish (EF)**

| **Task** | **Duration** | **ES** | **EF** | **Dependency** |
| --- | --- | --- | --- | --- |
| Define project scope | 2 days | 0 | 2 | - |
| Create project schedule | 3 days | 2 | 5 | Define project scope |
| Establish budget | 3 days | 5 | 8 | Create project schedule |
| Team assembly | 2 days | 8 | 10 | Establish budget |
| User research | 5 days | 10 | 15 | Team assembly |
| Visual design | 5 days | 15 | 20 | User research |
| Develop design guidelines | 5 days | 20 | 25 | Visual design |
| Android development | 10 days | 25 | 35 | Develop design guidelines |
| iOS development | 7 days | 35 | 42 | Android development |
| Integrate with backend API | 3 days | 42 | 45 | iOS development |
| API development | 8 days | 25 | 33 | Develop design guidelines |
| Database development | 4 days | 33 | 37 | API development |
| Conduct API testing | 3 days | 37 | 40 | Database development |
| Functional testing | 4 days | 45 | 49 | Integrate with backend API |
| Performance testing | 4 days | 49 | 53 | Functional testing |
| Identify and report defects | 3 days | 53 | 56 | Performance testing |
| Plan deployment strategy | 3 days | 56 | 59 | Identify and report defects |
| Deploy app to app stores | 4 days | 59 | 63 | Plan deployment strategy |
| Conduct training sessions | 4 days | 63 | 67 | Deploy app to app stores |
| Evaluate project success | 4 days | 67 | 71 | Conduct training sessions |
| Identify areas for improvement | 4 days | 71 | 75 | Evaluate project success |
| Develop maintenance plan | 9 days | 75 | 84 | Identify areas for improvement |

**Table 2: Latest Finish (LF) and Latest Start (LS)**

| **Task** | **Duration** | **LF** | **LS** | **Float** |
| --- | --- | --- | --- | --- |
| Develop maintenance plan | 9 days | 90 | 81 | 0 |
| Identify areas for improvement | 4 days | 81 | 77 | 6 |
| Evaluate project success | 4 days | 77 | 73 | 6 |
| Conduct training sessions | 4 days | 73 | 69 | 2 |
| Deploy app to app stores | 4 days | 69 | 65 | 6 |
| Plan deployment strategy | 3 days | 65 | 62 | 6 |
| Identify and report defects | 3 days | 62 | 59 | 6 |
| Performance testing | 4 days | 59 | 55 | 6 |
| Functional testing | 4 days | 55 | 51 | 6 |
| Conduct API testing | 3 days | 51 | 48 | 8 |
| Database development | 4 days | 48 | 44 | 7 |
| API development | 8 days | 44 | 36 | 3 |
| Integrate with backend API | 3 days | 51 | 48 | 3 |
| iOS development | 7 days | 48 | 41 | 6 |
| Android development | 10 days | 41 | 31 | 6 |
| Develop design guidelines | 5 days | 31 | 26 | 1 |
| Visual design | 5 days | 26 | 21 | 1 |
| User research | 5 days | 21 | 16 | 1 |
| Team assembly | 2 days | 16 | 14 | 4 |
| Establish budget | 3 days | 14 | 11 | 4 |
| Create project schedule | 3 days | 11 | 8 | 4 |
| Define project scope | 2 days | 8 | 6 | 6 |

Note:

* **LF (Latest Finish)**: The latest date by which a task must be completed.
* **LS (Latest Start)**: The latest date by which a task can start.
* **Float**: The amount of time a task can be delayed without affecting the overall project duration.

Q4.2 **Identifying the Critical Path**

The **critical path** is the path with no float (slack). The critical path in this project is:

* Define project scope → Create project schedule → Establish budget → Team assembly → User research → Visual design → Develop design guidelines → Android development → iOS development → Integrate with backend API → Functional testing → Performance testing → Identify and report defects → Plan deployment strategy → Deploy app to app stores → Conduct training sessions → Evaluate project success → Identify areas for improvement → Develop maintenance plan.

This critical path has a duration of **84 days** and ends on **December 31, 2024**, meaning the project is projected to finish on time.