2. SOFTWARE DEVELOPMENT LIFE CYCLE

2.1 Process Model:

In this model the Scrum Model is used. The reason for choosing this model is that the requirement of this project is not stable. Moreover, the project is a large project which is suitable for using the scrum model.

Scrum provides a completely well-organized structure consisting of Pregame, Development, and Post game phase. Which is perfectly suited for our program, as we are looking to develop software which has complex functionalities, has potential to future changes in requirements, requires collaboration with the customers, requires proper documentation and teamwork, requires daily meeting, backlogging and collaboration of the team. Regarding all these factors, scrum is the most feasible option for our software.

Scrum is progressive due to it being change-aware and collaborative. As opposed to the Waterfall methodology, which is described by strict phases (of requirements, design, implementation, testing, and deployment) that progress in a linear fashion with no backward movement, Scrum can be modified even during development. This is very relevant to your project, for example, “Protect the Precious”, where there may be new demands. Sufficiency of changes that can be introduced at previous stages makes the Waterfall model inappropriate for changing the requirements for the project in scope. Likewise, the V-Model puts more stress on verification and validation through its focus on development and testing completed in parallel, it does not provide an iterative approach like the Scrum. It is worth mentioning that the V-Model is quite rigid and is best suited for systems of critical importance to safety or quality of deliverables but suffers from changes in requirements very often which is the case when dealing with design projects like yours. In contrast, Scrum embeds testing within all sprints, providing the developers with timely QA and allowing problems like false alarms and privacy issues in your application to be flagged early on. Compared to other iterative models like DSDM, XP, FDD, and Incremental Model, in Scrum there is more organization and collaboration. DSDM also has timeboxes and user involvement as its focus but has a disadvantage of giving utmost importance to time making feature prioritization a bit inflexible. Thanks to Scrum’s iterative nature of sprints and the product backlog, your team can perform activities that involve placing priority on essential features, such as installing the automated alerts and emergency calling features. XP also advocates regular technical development and timely delivery but such developments such as pair programming require a lot of discipline and may not be relevant to your line of work. FDD is iterative in nature, but it focuses a lot on completing the specified features without wider avenues of interaction with other processes or stakeholders as is the case in Scrum. The Incremental Model and Prototyping Model share similarities with Scrum in their iterative nature but differ in their approach. The Incremental Model delivers the software in parts, with each increment building onto the previous one. However, changes to earlier increments can be difficult, making them less adaptable than Scrum. The Prototyping Model, while excellent for projects with unclear requirements, focuses primarily on creating early prototypes for feedback. This approach may not provide the continuous improvement and structured team collaboration needed for your project. Finally, Scrum’s collaborative nature aligns perfectly with your team-based project structure. Daily stand-ups, sprint planning, and retrospectives promote teamwork and ensure

smooth communication across all contributors. This contrasts with the siloed work environments common in models like Waterfall or V-Model. By choosing Scrum, your team can deliver a dynamic, high-quality application that evolves in response to real-world needs, making it the ideal choice for "Protect the Precious."

2.2 Project Role Identification and Responsibility

i. Scrum Master

- **Role**: Guides the team and ensures Scrum rules are followed.
- **Responsibilities**:
  - Runs meetings like Sprint Planning and Daily Scrum.
  - Helps the team solve problems, like issues with live location or safety alerts.
  - Make sure the team stays on track.

ii. Product Owner

- **Role**: Manages the features and priorities for the app.
- **Responsibilities**:
  - Decides what to build first, like emergency alerts or child monitoring.
  - Updates the task list (Product Backlog) based on user feedback.
  - Works with parents, children, and stakeholders to define the app's needs.

iii. Scrum Team

- **Role**: Builds the app features and tests them.
- **Responsibilities**:
  - Develop features like map zones, panic buttons, and automated calls.
  - Fixes issues and deliver updates in short cycles (Sprints).
  - Collaborate daily to track progress and solve issues.

iv. Stakeholders

- **Role**: Give feedback and support the project.
- **Responsibilities**:
  - Parents and children test the app and improvements.
  - Management approves resources and support development.