Cleveland State University
CIS 434 — Software Engineering

Project Plan
GiveOne

Submitted by:
Ahmed Shady

Date:
September 16, 2025

1.0 Introduction

1.1 Project Scope

GiveOne is a donation app that makes giving easy and meaningful by allowing users to contribute just $1 per day to real cases posted by organizations. The app tracks progress, celebrates milestones, and delivers final updates from organizations once cases are funded and completed.

- **Inputs:** User donations ($1 increments), organization case details.
- **Processing:** Wallet balance updates, progress tracking, case status changes (Open → Funded → Completed).
- **Outputs:** Progress bars, donation history, updates from organizations.

1.2 Major Software Functions

- Allow users to donate $1/day to a case of their choice.
- Show progress toward funding goals.
- Mark cases as funded once goals are met.
- Allow organizations to post final updates.
- (Future) Daily reminders, auto-donations, dashboards.

1.3 Performance/Behavior Issues

- Must respond quickly when donations are made.
- Must store data persistently between sessions.
- Should be easy to use for non-technical users.

1.4 Management and Technical Constraints

- Solo project (Ahmed Shady).
- Limited resources and time.
- Technology: Python with Tkinter for desktop app, JSON/SQLite for persistence.

2.0 Risk Management

2.1 Project Risks

- **Scope creep**: Adding too many features may delay MVP.
- **Time management**: Solo project risks falling behind schedule.
- **Technical complexity**: Some features (auto-donate, reminders) may require extra learning.

2.2 Risk Table

Risk	Probability	Impact	Mitigation
Scope creep	Medium	High	Focus on MVP first.
Time management	High	High	Stick to weekly milestones.
Technical complexity	Medium	Medium	Use simple tools (Tkinter + JSON).

2.3 Risk Mitigation, Monitoring, and Management (RM3)

- Weekly self-check-ins to monitor progress.
- Keep non-MVP features as "future work."
- If blocked, simplify functionality to stay on track.

3.0 Project Schedule

3.1 Project Task Set

- Requirements & SRS (Weeks 1–2).
- Design & diagrams (Weeks 3–4).
- Build MVP (Weeks 5–6).
- Test plan + testing (Week 7).
- Extra features + documentation (Weeks 8–10).

3.2 Functional Decomposition

- **Donations:** Handle $1 contributions.
- **Wallet:** Track and update balance.
- **Cases:** Track progress, status, and updates.
- **History:** Keep record of all donations.

3.3 Task Network

Requirements → Design → Development → Testing → Documentation → Presentation.

3.4 Timeline Chart

- **Sept 16:** Project Plan.
- **Sept 30:** SRS.
- **Oct 14:** Design Spec + 2-min Presentation.
- **Oct 28:** Initial Demo.
- **Nov 4:** Test Plan.
- **Dec 2–4:** Final Presentation.
- **Dec 4:** Final Deliverables.

3.5 Schedule Compliance

Tasks will be tracked weekly and checked against course deadlines.

4.0 Roles & Responsibilities

Since this is a solo project, Ahmed Shady will complete all project tasks including requirements, design, coding, testing, documentation, and the final presentation/demo.

5.0 Appendix

- Future Work: daily reminders, auto-donations, organization dashboards.
- Tools: Python, Tkinter, JSON/SQLite.
- Author: Ahmed Shady.