OOP and Design Patterns (CSCI 375)
Student Showcase (Final Project) Rubric

1. Project Title: Bloom-Affirmation App

2. Team Members: Lauren Lewis and Ashley Velasquez

3. Evaluator:

Instructions:

1. There are 10 technical requirements (worth 100 points) to grade the project and the team presentation.
2. For each requirement, use 0 - 10 scale in the Score column (0-6: Fail, 7: C (Average), 8: B (Above Average), 9: A (Good), (10: Excellent)
3. Use the *Notes* section to jot down any observations that may help in grading and justification.

Team and Technical Project Requirement	**Score**
1. Use **4+1 Views** to explain the software design to the audience. Notes: Lauren completed the logical, process, and physical view while Ashley did the context and development view	10/10
2. **Use of 3 Design Patterns** -- presentation clearly stated and briefly explained design patterns use. Common design patterns are Iterator, Decorator, Observer, Strategy, Command, State, Singleton, Adapter, Façade, Flyweight, Abstract Factory, Composite, Template, MVC, etc. Notes: We used many patterns including Singleton DB connections, repositories and services, Factory Methods used for sprouts/users, and template behavior used in our domain model	10/10
3. **Use of Fundamental OOD Concepts-** e.g., Inheritance, Abstraction, Getters and Setters, Overloading, Attributes and Methods, etc. Notes: The project demonstrates strong use of core OOP concepts including abstraction and inheritance in the domain models, encapsulation through getters and setter and method overloading. Attributes and methods were kept well organized and consisted across modules	10/10
4. **Software management** – good usage of management, communication and tracking tools e.g., Gant chart, Kanban board, GitHub Project, Clickup, Discord, Slack, etc.	8/10

Notes: Although we struggled with Test-First development and version control on GitHub, we adapted and the project progressed. We used Discord and SMS messaging to coordinate work and share updates. In the end we created two repos, one that got messy during development and a cleaned-up version we submitted our final on.	
5. **Documentation** – clear, easy to follow documentation, UML diagrams are complete, and notations are correct; explanation of objects interaction is clear and complete. Notes: All UML diagrams were completed with correct notations and documentation thoroughly explained class relationships and object interaction	10/10
6. **Testing and CI-CD pipeline – automatically generates test data** using hypothesis, usage of mocking/patching, provides code coverage and Python type check (mypy) reports, CI-CD, Docker, etc. Notes: Tests were implemented using mocking and patching techniques. The project uses type checking (mypy) and follows suitable for CI/CD.	10 /10
7. **Clean code and Modularity** – project uses appropriate software and system design; the code follows best practices (pep8), is well organized and refactored into packages, modules, classes, etc. Notes: Our code follows strong modular design. Code is organized into packages, classes and modules that each have a responsibility. The final structure is easy to navigate and extend.	10/10
8. **Project requirements and execution** -- clearly stated functional and technical requirements, project adequately challenging for sophomore-junior students; project demo was clear and concise, etc. Notes: The project met all functional requirements and was appropriately challenging.	10/10
9. **Team presentation** -- all members participated in presentation, used the visual and oral presentation techniques and tools to engage audience, etc. Notes: Both members presented and contributed to explaining the technical and design components	10/10
10. **Individual Contributions** – briefly explain how and what each member contributed to the successful execution of the project. Notes:	10/10

Ashley primarily focused on the Domain layer including Base Models and writing associated unit test, contributing when necessary. Lauren led development across most remaining modules and coordinated UI integration.	
11. **BONUS: Above and beyond** – Team went beyond the above list e.g., great User Interface, use of Database, security, real-world application, real-world client delight and interaction, etc. Notes: Lauren created a functional user interface using an emulator to demonstrate how our project was completed and simulate our working app.	10/10
Total Score Note: Max score can be 110 due to 10 BONUS points.	108/100