CIS 440 Team 2

Sereena Mounce (Developer)
Aditya Dabberu (Developer)
Srishti Garg (Developer)
Wesley Helm (Developer)
Kayla Sermini (Scrum Master)
Natalie Alexander (Product Owner)

1. Executive Summary

Self-Care Journal is a web application designed to promote mental well-being and personal growth through digital journaling and self-reflection. This POC outlines the core features, technology stack, and development timeline for the initial version of the application.

2. Key Features

1. User Authentication
2. Daily Reflection Journal
3. Gratitude Log
4. Goal Setting and Progress Monitoring

3. Technology Stack

- Backend: Flask (Python)
- Frontend: HTML, CSS, JavaScript, Bootstrap
- Database: MySQL
- Cloud Platform: Azure
- Additional Tools: Flask-SQLAlchemy, Flask-Login

4. Architecture Overview

The application follows a Model-View-Controller (MVC) architecture using Flask:

- Models: Defined using SQLAlchemy ORM
- Views: HTML templates with Jinja2
- Controllers: Flask routes and business logic

Project Structure:

```
/accountify
├── run.py # Main file to start the Flask app
├── app.py # Backend logic and routes
├── requirements.txt # Dependencies for the project
├── templates/ # HTML files for the frontend
│ ├── login.html # Login page
│ ├── dashboard.html # Main dashboard (includes journal, gratitude, and goals)
│ ├── register.html # Registration page
├── static/ # Static assets (CSS, JS)
│ ├── styles.css # Custom styles
│ ├── index.js # Main JavaScript file for frontend logic
├── models.py # Database models
```

5. Feature Details and Sequence Diagrams

5.1 User Authentication

```mermaid
sequenceDiagram
    participant User
    participant Browser
    participant Server
    participant Database

    User->>Browser: Enter credentials
    Browser->>Server: POST /login
    Server->>Database: Verify credentials
    Database-->>Server: Credentials valid
    Server-->>Browser: Set session cookie
    Browser-->>User: Redirect to dashboard
```

5.2 Daily Reflection Journal

text
sequenceDiagram
    participant User
    participant Browser
    participant Server
    participant Database

    User->>Browser: Write journal entry
    Browser->>Server: POST /journal/new
    Server->>Database: Save entry
    Database-->>Server: Entry saved
    Server-->>Browser: Confirmation
    Browser-->>User: Display success message

5.3 Gratitude Log

text
sequenceDiagram
    participant User
    participant Browser
    participant Server
    participant Database

    User->>Browser: Add gratitude item
    Browser->>Server: POST /gratitude/add
    Server->>Database: Save gratitude item
    Database-->>Server: Item saved
    Server-->>Browser: Updated gratitude list
    Browser-->>User: Display updated list

6. Database Schema

sql
```sql
CREATE TABLE users (
    id INT PRIMARY KEY AUTO_INCREMENT,
    username VARCHAR(50) UNIQUE NOT NULL,
    email VARCHAR(120) UNIQUE NOT NULL,
    password_hash VARCHAR(128) NOT NULL,
```

```
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE journal_entries (
    id INT PRIMARY KEY AUTO_INCREMENT,
    user_id INT,
    content TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(id)
);

CREATE TABLE gratitude_logs (
    id INT PRIMARY KEY AUTO_INCREMENT,
    user_id INT,
    content TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(id)
);

CREATE TABLE goals (
    id INT PRIMARY KEY AUTO_INCREMENT,
    user_id INT,
    title VARCHAR(100),
    description TEXT,
    status VARCHAR(20),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(id)
);
```

7. API Endpoints

- POST /auth/register
- POST /auth/login
- POST /auth/logout
- GET /journal
- POST /journal/new
- GET /gratitude
- POST /gratitude/add
- GET /goals

- POST /goals/new
- PUT /goals/<id>/update

8. Development Timeline

Week 1: Planning & Setup

- Define user stories and tasks in Trello
- Set up GitHub repository
- Create database schema
- Build static HTML/CSS pages for login, journal, and dashboard

Week 2: Core Development

- Implement user authentication
- Develop journal entry and gratitude log features
- Create goal setting and tracking functionality
- Integrate frontend with backend APIs

Week 3: Integration & Testing

- Implement data visualization for journal entries and goals
- Conduct thorough testing of all features
- Optimize performance and fix bugs
- Prepare documentation and demo script

9. Risks and Mitigation Strategies

1. Data Security
    - Risk: Unauthorized access to user data
    - Mitigation: Implement robust authentication, use HTTPS, and encrypt sensitive data
2. Scalability
    - Risk: Performance issues with increased user base
    - Mitigation: Design for scalability, use caching, and optimize database queries
3. User Adoption
    - Risk: Low user engagement
    - Mitigation: Implement intuitive UI/UX and gamification elements

10. Future Enhancements

1. Mobile application for iOS and Android
2. Integration with wearable devices for mood tracking
3. AI-powered journaling prompts and insights
4. Social features for community support
5. Integration with professional therapy services

11. Conclusion

The Self-Care Journal application provides a solid foundation for users to engage in daily reflection, practice gratitude, and set personal goals. With the outlined features and development plan, we are well-positioned to create a valuable tool for promoting mental well-being and personal growth.