Project Title

Smart Exercise & Fitness Tracker Web App

Project Description

The **Smart Exercise & Fitness Tracker** is a React + Python web application that helps users manage their fitness journey effectively. The app provides personalized workout routines, tailored diet plans, and a video tutorial library. It integrates with Firebase for authentication and data storage while also including an AI-powered chatbot to answer fitness-related queries.

The system focuses on security (auth + sharing rules), privacy (consent & audit logs), and testable features with clear acceptance criteria, ensuring that the application can serve as a reliable and interactive fitness companion.

Tools & Technologies

- **Frontend:** React.js, Bootstrap / Material UI
- **Backend:** Python (Flask or FastAPI)
- **Database & Auth:** Firebase (Firestore + Firebase Auth)
- **AI / ML:** scikit-learn, TensorFlow / external API for calorie estimation
- **Other Tools:** GitHub (version control), Figma (UI design), Postman (API testing)

Feature List (with Acceptance Criteria)

1. Natural-language exercise query (Chat & HTML output)

- User can type queries like *"Show me exercises for this week"*.
- Backend parses query → fetches data from DB → returns **HTML-formatted response** (exercise name, description, sets/reps, duration, optional image).

  ➔ **Acceptance Criteria:**

  - Query returns only user's exercises for the relevant week.
  - Each row includes <img> if available, otherwise placeholder.
  - HTML is valid and safely rendered in frontend chat view.

---

2. Dashboard, history & recommended queries

- Responsive dashboard showing daily/weekly progress (exercise completion, calories).
- History log of last 30 days.
- Recommended queries: auto-suggest 3 based on user behavior.

  ➔ **Acceptance Criteria:**

- Dashboard filters by date range.
- Export (CSV) button works.
- Recommendations update after repetitive queries or new usage patterns.

---

3. Daily food image capture & calorie estimation

- User uploads/captures food images.
- Backend pipeline (ML model / API) estimates calories.
- If low confidence, system shows top 3 guesses and asks user to confirm.

➔ **Acceptance Criteria:**

- Stores image, detected food, calorie estimate, and confirmation in DB.
- Accuracy documented with expected error bounds.
- User prompted if confidence < 0.6.

---

4. Authentication, sessions & secure sharing

- Login/register with Firebase Auth.
- Sessions via JWT or Firebase tokens.
- Users may share diet/exercise plans with specific others (view-only or view+comment).

➔ **Acceptance Criteria:**

- Protected APIs require authentication.
- Shared resources visible only to users in ACL.
- Session invalidates on logout or revoke.

---

5. Logging, audit & privacy

- All image uploads & sharing actions logged with timestamp + user ID.
- Consent prompt before first image upload.
- Privacy page explains data usage, retention, and sharing rules.

➔ **Acceptance Criteria:**

- Audit log entries recorded in DB.
- User cannot upload images without accepting privacy terms.

---

6. Admin / QA endpoints

- Admin can review flagged images, errors, or reported issues.

➔ **Acceptance Criteria:**

- Admin dashboard shows flagged items.
- Admin can mark issues as resolved.

Real-Life Problem Solved

Many individuals struggle with consistency, planning, and guidance in fitness. Our solution provides:

- **Personalized planning** (diet & exercise)
- **Motivation & accountability** (progress tracking, chatbot assistance)
- **Data-driven insights** (dashboard & history)
- **Secure sharing** with friends/trainers

Thus, it bridges the gap between professional fitness coaching and self-management.

Team Members

1. Bansi Vachhani

   En No: 22012011050

   Email: bansivachhani153@gmail.com

   Phone: 8780762365

2. Bhakti Kansagara

   En No: 22012011065

   Email: bhaktikansagara2004@gmail.com

   Phone: 9909227175

3. Hill Soni

   En No: 22012011048

   Email: hillsoni8104@gmail.com

   Phone: 9429192301

Division of Work (Module-based)

- **Member A (Backend Lead — Auth & Database):**
  Firebase Auth, Firestore schema, CRUD APIs, sharing ACL, sessions, audit logging.
- **Member B (AI/ML Lead — Chatbot & Recommendations):**
  Food image recognition, calorie estimation, chatbot parsing, recommendation engine.

- **Member C (Frontend Lead — React UI & UX):**
  Chat interface, dashboard, image upload UI, sharing/consent forms, responsive design.

Weekly Work Plan (14 Weeks)

Week	Member A (Auth/DB)	Member B (AI/Chatbot)	Member C (Frontend)
1-2	Firebase setup, schema	Collect datasets, chatbot rules	Figma wireframes, React setup
3-4	Auth APIs + CRUD	Basic chatbot (rule-based)	Login/Signup UI
5-6	Exercise/diet APIs	Train calorie-estimate model	Dashboard UI
7-8	Progress tracker API	Chatbot integration	API connections
9-10	Notifications API	NLP improvements	Video tutorials UI
11-12	Audit + privacy module	Fine-tune chatbot	UI polishing
13	Integration testing	Integration testing	Integration testing
14	Docs & final QA	Docs & final QA	Deployment

Repository / File Links

GitHub: [https://github.com/bansivachhani/Smart-Exercise-Fitness-Tracker]

Proposed Timeline & Milestones

- **Start Date:** 19 August 2025
- **Week 2:** Firebase auth & basic UI completed
- **Week 6:** Exercise/diet APIs + chatbot prototype ready
- **Week 10:** Full app integrated with chatbot & video tutorials
- **Week 13:** Final integration & QA complete
- **Demo Date:** 25 November 2025
- **Final Submission:** 30 November 2025

Hardware / Special Resources

- Laptops with Python & Node.js
- Firebase account (free tier)
- Fitness datasets (Kaggle / open source)
- Stable internet connection

Risks & Mitigation

- **Firebase integration issues** → small prototypes first
- **ML model accuracy** → start rule-based, improve with ML
- **Integration delays** → agree on API contracts early + use Postman

---

API Usage Examples

API Usage Examples (Postman / cURL)

Example 1: Fetch User Exercises

Request:

```
curl -X GET "http://localhost:5000/api/exercises?user_id=123" \
    -H "Authorization: Bearer <token>"
```

Expected Response:

```
[
  {
    "exercise": "Push Ups",
    "sets": 3,
    "reps": 15,
    "duration": "10 min"
  }
]
```

Example 2: Upload Food Image for Calorie Estimation

Request:

```
curl -X POST "http://localhost:5000/api/food" \
    -F "image=@food.jpg" \
    -H "Authorization: Bearer <token>"
```

Expected Response:

```
{
```

"food": "Pasta",
"calories": 340,
"confidence": 0.82
}

Testing Checklist

☑ Login & Signup works with Firebase Authentication

☑ Dashboard shows daily/weekly progress correctly

☑ Food image upload estimates calories and stores data

☑ Privacy consent is required before uploading any image

☑ Sharing feature only works for users in ACL (view-only or comment access)

☑ Audit logs are created for uploads and sharing actions

☑ Admin dashboard shows flagged content and allows resolution

Academic Integrity Declaration

We declare this project as our original work. Datasets, libraries, or reused code will be cited properly. AI tools (e.g., ChatGPT) have been used for brainstorming and debugging, **not** for final deliverables.