Test Cases

Gym Management System

1. TEST CASE OVERVIEW

Total Test Cases: 50+
Test Categories:

- Unit Tests (Authentication, Services)
- Integration Tests (API Endpoints)
- Database Tests (CRUD Operations)
- Security Tests (Authentication, Authorization)
- Validation Tests (Input Validation)

2. AUTHENTICATION MODULE TEST CASES

2.1 Member Registration Tests

TC-AUTH-001: Successful Member Registration

Test Case ID	TC-AUTH-001
Module	Authentication
Test Scenario	Register a new member with valid data
Prerequisites	Database is accessible
Test Data	name: "John Doe" email: "john@test.com" password: "Test@123" age: 25 gender: "Male" phone: "1234567890"
Test Steps	1. Send POST request to `/api/auth/signup` 2. Include all required fields 3. Verify response

Test Case ID	TC-AUTH-001
Expected Result	- HTTP Status: 201 Created - Response contains member details - Access token generated - Refresh token generated - Password is hashed in database
Actual Result	As expected
Status	✅ Pass

TC-AUTH-002: Registration with Duplicate Email

Test Case ID	TC-AUTH-002
Module	Authentication
Test Scenario	Attempt to register with existing email
Prerequisites	Member with email "john@test.com" already exists
Test Data	email: "john@test.com" (other valid data)
Test Steps	1. Send POST request to `/api/auth/signup` 2. Use duplicate email 3. Verify error response
Expected Result	- HTTP Status: 400 Bad Request - Error message about duplicate email
Actual Result	As expected
Status	✅ Pass

TC-AUTH-003: Registration with Missing Fields

Test Case ID	TC-AUTH-003
Module	Authentication
Test Scenario	Register without required fields
Prerequisites	Database is accessible
Test Data	name: "John Doe" email: "john@test.com" (missing password, age, gender, phone)

Test Case ID	TC-AUTH-003
Test Steps	1. Send POST request with incomplete data 2. Verify validation error
Expected Result	- HTTP Status: 400 Bad Request - Validation error message
Actual Result	As expected
Status	✅ Pass

TC-AUTH-004: Registration with Invalid Email Format

Test Case ID	TC-AUTH-004
Module	Authentication
Test Scenario	Register with invalid email format
Prerequisites	Database is accessible
Test Data	email: "invalid-email" (other valid data)
Test Steps	1. Send POST request with invalid email 2. Verify validation error
Expected Result	- HTTP Status: 400 Bad Request - Email format validation error
Actual Result	As expected
Status	✅ Pass

TC-AUTH-005: Registration with Weak Password

Test Case ID	TC-AUTH-005
Module	Authentication
Test Scenario	Register with weak password (if validation implemented)
Prerequisites	Database is accessible
Test Data	password: "123" (other valid data)
Test Steps	1. Send POST request with weak password 2. Verify validation error

Test Case ID	TC-AUTH-005
Expected Result	- HTTP Status: 400 Bad Request - Password strength error
Actual Result	Depends on validation rules
Status	⚠️ Optional

2.2 Member Login Tests

TC-AUTH-006: Successful Member Login

Test Case ID	TC-AUTH-006
Module	Authentication
Test Scenario	Login with valid credentials
Prerequisites	Member exists in database
Test Data	email: "john@test.com" password: "Test@123"
Test Steps	1. Send POST request to `/api/auth/login` 2. Provide valid credentials 3. Verify response
Expected Result	- HTTP Status: 200 OK - Response contains member details - Access token generated - Refresh token generated
Actual Result	As expected
Status	✅ Pass

TC-AUTH-007: Login with Incorrect Password

Test Case ID	TC-AUTH-007
Module	Authentication
Test Scenario	Login with wrong password
Prerequisites	Member exists in database

Test Case ID	TC-AUTH-007
Test Data	email: "john@test.com" password: "WrongPassword"
Test Steps	1. Send POST request with incorrect password 2. Verify error response
Expected Result	- HTTP Status: 401 Unauthorized - Error: "Invalid credentials"
Actual Result	As expected
Status	✅ Pass

TC-AUTH-008: Login with Non-existent Email

Test Case ID	TC-AUTH-008
Module	Authentication
Test Scenario	Login with email not in database
Prerequisites	Email does not exist
Test Data	email: "nonexistent@test.com" password: "Test@123"
Test Steps	1. Send POST request with non-existent email 2. Verify error response
Expected Result	- HTTP Status: 401 Unauthorized - Error: "Invalid credentials"
Actual Result	As expected
Status	✅ Pass

2.3 Trainer Login Tests

TC-AUTH-009: Successful Trainer Login

Test Case ID	TC-AUTH-009
Module	Authentication
Test Scenario	Trainer login with valid credentials
Prerequisites	Trainer exists (seeded)

Test Case ID	TC-AUTH-009
Test Data	email: "john@gym.com" password: "trainer123"
Test Steps	1. Send POST request to `/api/auth/trainer/login` 2. Verify response
Expected Result	- HTTP Status: 200 OK - Response contains trainer details - Tokens generated with role='trainer'
Actual Result	As expected
Status	✅ Pass

TC-AUTH-010: Trainer Login with Wrong Password

Test Case ID	TC-AUTH-010
Module	Authentication
Test Scenario	Trainer login with incorrect password
Prerequisites	Trainer exists in database
Test Data	email: "john@gym.com" password: "wrongpass"
Test Steps	1. Send POST request with incorrect password 2. Verify error response
Expected Result	- HTTP Status: 401 Unauthorized - Error: "Invalid credentials"
Actual Result	As expected
Status	✅ Pass

---

3. MEMBER MODULE TEST CASES

3.1 Member Profile Tests

TC-MEMBER-001: Get Member Profile (Authenticated)

Test Case ID	TC-MEMBER-001
Module	Member
Test Scenario	Authenticated member retrieves their profile
Prerequisites	Member is logged in with valid token
Test Data	Authorization: Bearer {valid_token}
Test Steps	1. Send GET request to `/api/member/profile` 2. Include auth token in header 3. Verify response
Expected Result	- HTTP Status: 200 OK - Response contains member profile data - Password is not included in response
Actual Result	As expected
Status	✅ Pass

TC-MEMBER-002: Get Profile Without Token

Test Case ID	TC-MEMBER-002
Module	Member
Test Scenario	Access profile without authentication
Prerequisites	None
Test Data	No authorization header
Test Steps	1. Send GET request without token 2. Verify error response
Expected Result	- HTTP Status: 401 Unauthorized - Error: "No token provided"
Actual Result	As expected
Status	✅ Pass

TC-MEMBER-003: Get Profile with Invalid Token

Test Case ID	TC-MEMBER-003
Module	Member
Test Scenario	Access profile with invalid token

Test Case ID	TC-MEMBER-003
Prerequisites	None
Test Data	Authorization: Bearer invalid_token
Test Steps	1. Send GET request with invalid token 2. Verify error response
Expected Result	- HTTP Status: 401 Unauthorized - Error: "Invalid or expired token"
Actual Result	As expected
Status	✅ Pass

TC-MEMBER-004: Get Profile with Expired Token

Test Case ID	TC-MEMBER-004
Module	Member
Test Scenario	Access profile with expired token
Prerequisites	Token has expired (after 15 minutes)
Test Data	Authorization: Bearer {expired_token}
Test Steps	1. Wait for token to expire 2. Send GET request 3. Verify error
Expected Result	- HTTP Status: 401 Unauthorized - Error: "Invalid or expired token"
Actual Result	As expected
Status	✅ Pass

3.2 Workout Plans Tests

TC-MEMBER-005: Get My Workout Plans

Test Case ID	TC-MEMBER-005
Module	Member
Test Scenario	Member retrieves assigned workout plans

Test Case ID	TC-MEMBER-005
Prerequisites	Member is authenticated and has workout plans
Test Data	Authorization: Bearer {valid_member_token}
Test Steps	1. Send GET request to `/api/member/my/workout` 2. Verify response
Expected Result	- HTTP Status: 200 OK - Array of workout plans - Each plan includes trainer info
Actual Result	As expected
Status	✅ Pass

TC-MEMBER-006: Get Workout Plans (No Plans)

Test Case ID	TC-MEMBER-006
Module	Member
Test Scenario	Member has no workout plans
Prerequisites	Member has no assigned plans
Test Data	Authorization: Bearer {valid_member_token}
Test Steps	1. Send GET request 2. Verify empty array returned
Expected Result	- HTTP Status: 200 OK - Empty array []
Actual Result	As expected
Status	✅ Pass

3.3 Diet Plans Tests

TC-MEMBER-007: Get My Diet Plans

Test Case ID	TC-MEMBER-007
Module	Member
Test Scenario	Member retrieves assigned diet plans

Test Case ID	TC-MEMBER-007
Prerequisites	Member is authenticated
Test Data	Authorization: Bearer {valid_member_token}
Test Steps	1. Send GET request to `/api/member/my/diet` 2. Verify response
Expected Result	- HTTP Status: 200 OK - Array of diet plans - Includes trainer details
Actual Result	As expected
Status	✅ Pass

3.4 Attendance Tests

TC-MEMBER-008: Get My Attendance

Test Case ID	TC-MEMBER-008
Module	Member
Test Scenario	Member retrieves attendance history
Prerequisites	Member is authenticated
Test Data	Authorization: Bearer {valid_member_token}
Test Steps	1. Send GET request to `/api/member/my/attendance` 2. Verify response
Expected Result	- HTTP Status: 200 OK - Array of attendance records - Sorted by date (desc)
Actual Result	As expected
Status	✅ Pass

3.5 Progress Tests

TC-MEMBER-009: Get My Progress

Test Case ID	TC-MEMBER-009
Module	Member
Test Scenario	Member retrieves progress records
Prerequisites	Member is authenticated
Test Data	Authorization: Bearer {valid_member_token}
Test Steps	1. Send GET request to `/api/member/my/progress` 2. Verify response
Expected Result	- HTTP Status: 200 OK - Array of progress records - Includes weight, body_fat, muscle_mass - Includes trainer name
Actual Result	As expected
Status	✅ Pass

---

4. TRAINER MODULE TEST CASES

4.1 View Members Tests

TC-TRAINER-001: Get All Members (Trainer)

Test Case ID	TC-TRAINER-001
Module	Trainer
Test Scenario	Trainer retrieves all gym members
Prerequisites	Trainer is authenticated
Test Data	Authorization: Bearer {valid_trainer_token}
Test Steps	1. Send GET request to `/api/trainer/members` 2. Verify response
Expected Result	- HTTP Status: 200 OK - Array of all members - Passwords not included
Actual Result	As expected
Status	✅ Pass

TC-TRAINER-002: Get Members (Member Token)

Test Case ID	TC-TRAINER-002
Module	Trainer
Test Scenario	Member tries to access trainer-only endpoint
Prerequisites	Member is authenticated
Test Data	Authorization: Bearer {valid_member_token}
Test Steps	1. Send GET request to `/api/trainer/members` 2. Verify error
Expected Result	- HTTP Status: 403 Forbidden - Error: "Insufficient permissions"
Actual Result	As expected
Status	✅ Pass

4.2 Assign Workout Plan Tests

TC-TRAINER-003: Assign Workout Plan (Success)

Test Case ID	TC-TRAINER-003
Module	Trainer
Test Scenario	Trainer assigns workout plan to member
Prerequisites	Trainer and member exist

Test Data	Authorization: Bearer {valid_trainer_token}
	Body: { plan_details: "5 days strength training" }

Test Steps	1. Send PUT request to `/api/trainer/members/:id/workout` 2. Include plan details 3. Verify response
Expected Result	- HTTP Status: 200 OK - Workout plan created - Plan linked to member and trainer
Actual Result	As expected
Status	✅ Pass

TC-TRAINER-004: Assign Plan (Invalid Member ID)

Test Case ID	TC-TRAINER-004
Module	Trainer
Test Scenario	Assign plan to non-existent member
Prerequisites	Trainer is authenticated
Test Data	Member ID: "invalid-uuid" Valid plan details
Test Steps	1. Send PUT request with invalid member ID 2. Verify error
Expected Result	- HTTP Status: 404 Not Found or 400 Bad Request - Error about invalid member
Actual Result	As expected
Status	✅ Pass

TC-TRAINER-005: Assign Plan (Member Token)

Test Case ID	TC-TRAINER-005
Module	Trainer
Test Scenario	Member tries to assign workout plan
Prerequisites	Member is authenticated
Test Data	Authorization: Bearer {valid_member_token}
Test Steps	1. Send PUT request to trainer endpoint 2. Verify error
Expected Result	- HTTP Status: 403 Forbidden - Error: "Insufficient permissions"
Actual Result	As expected
Status	✅ Pass

4.3 Assign Diet Plan Tests

TC-TRAINER-006: Assign Diet Plan (Success)

Test Case ID	TC-TRAINER-006
Module	Trainer
Test Scenario	Trainer assigns diet plan to member
Prerequisites	Trainer and member exist
Test Data	Authorization: Bearer {valid_trainer_token} Body: { diet_details: "High protein low carb" }
Test Steps	1. Send PUT request to `/api/trainer/members/:id/diet` 2. Verify response
Expected Result	- HTTP Status: 200 OK - Diet plan created and linked
Actual Result	As expected
Status	✅ Pass

TC-TRAINER-007: Assign Diet (Missing Details)

Test Case ID	TC-TRAINER-007
Module	Trainer
Test Scenario	Assign diet without plan details
Prerequisites	Trainer is authenticated
Test Data	Body: {} (empty)
Test Steps	1. Send PUT request without diet_details 2. Verify validation error
Expected Result	- HTTP Status: 400 Bad Request - Validation error
Actual Result	As expected
Status	✅ Pass

4.4 Record Attendance Tests

TC-TRAINER-008: Record Attendance (Success)

Test Case ID	TC-TRAINER-008
Module	Trainer
Test Scenario	Trainer records member attendance
Prerequisites	Trainer and member exist

Test Data	Authorization: Bearer {valid_trainer_token}
	Body: { status: "present" }

Test Steps	1. Send POST request to `/api/trainer/members/:id/attendance` 2. Verify response
Expected Result	- HTTP Status: 201 Created - Attendance record created
Actual Result	As expected
Status	✅ Pass

TC-TRAINER-009: Record Attendance (Absent)

Test Case ID	TC-TRAINER-009
Module	Trainer
Test Scenario	Mark member as absent
Prerequisites	Trainer and member exist
Test Data	Body: { status: "absent" }
Test Steps	1. Send POST request with status="absent" 2. Verify record created
Expected Result	- HTTP Status: 201 Created - Attendance with status "absent"
Actual Result	As expected
Status	✅ Pass

4.5 Update Progress Tests

TC-TRAINER-010: Update Progress (Success)

Test Case ID	TC-TRAINER-010
Module	Trainer
Test Scenario	Trainer updates member progress
Prerequisites	Trainer and member exist

Test Data	Authorization: Bearer {valid_trainer_token}
	Body: { weight: 75.5, body_fat: 18.2, muscle_mass: 35.0, notes: "Good progress" }

Test Steps	1. Send PUT request to `/api/trainer/members/:id/progress` 2. Verify response
Expected Result	- HTTP Status: 200 OK - Progress record created - All metrics saved
Actual Result	As expected
Status	✅ Pass

TC-TRAINER-011: Update Progress (Missing Fields)

Test Case ID	TC-TRAINER-011
Module	Trainer
Test Scenario	Update progress without required fields
Prerequisites	Trainer is authenticated
Test Data	Body: { weight: 75.5 } (missing body_fat, muscle_mass)
Test Steps	1. Send PUT request with incomplete data 2. Verify validation error
Expected Result	- HTTP Status: 400 Bad Request - Validation error about missing fields
Actual Result	As expected
Status	✅ Pass

TC-TRAINER-012: Update Progress (Negative Values)

Test Case ID	TC-TRAINER-012
Module	Trainer

Test Case ID	TC-TRAINER-012
Test Scenario	Update progress with invalid negative values
Prerequisites	Trainer is authenticated
Test Data	Body: { weight: -75, body_fat: -10, muscle_mass: -20 }
Test Steps	1. Send PUT request with negative values 2. Verify validation error
Expected Result	- HTTP Status: 400 Bad Request - Validation error about invalid values
Actual Result	Depends on validation implementation
Status	⚠️ Should Implement

---

5. DATABASE TEST CASES

5.1 CRUD Operations Tests

TC-DB-001: Create Member Record

Test Case ID	TC-DB-001
Module	Database
Test Scenario	Insert new member into database
Test Steps	1. Use Prisma to create member 2. Verify record exists 3. Check all fields
Expected Result	- Record created with UUID - All fields stored correctly - Timestamps auto-generated
Status	✅ Pass

TC-DB-002: Read Member Record

Test Case ID	TC-DB-002
Module	Database
Test Scenario	Retrieve member by ID

Test Case ID	TC-DB-002
Test Steps	1. Query member by member_id 2. Verify data returned
Expected Result	- Correct member data retrieved - No password in select query
Status	✅ Pass

TC-DB-003: Update Member Record

Test Case ID	TC-DB-003
Module	Database
Test Scenario	Update member information
Test Steps	1. Update member phone number 2. Verify update persisted
Expected Result	- Update successful - Only specified fields changed
Status	✅ Pass

TC-DB-004: Delete Member (Cascade)

Test Case ID	TC-DB-004
Module	Database
Test Scenario	Delete member and verify cascade
Test Steps	1. Create member with plans 2. Delete member 3. Verify related records deleted
Expected Result	- Member deleted - Workout plans deleted - Diet plans deleted - Attendance deleted - Progress deleted
Status	✅ Pass

5.2 Constraint Tests

TC-DB-005: Unique Email Constraint

Test Case ID	TC-DB-005
Module	Database
Test Scenario	Attempt to insert duplicate email
Test Steps	1. Insert member with email 2. Try to insert another with same email
Expected Result	- First insert succeeds - Second insert fails with unique constraint error
Status	✅ Pass

TC-DB-006: Foreign Key Constraint

Test Case ID	TC-DB-006
Module	Database
Test Scenario	Create workout plan with invalid member_id
Test Steps	1. Try to create plan with non-existent member_id
Expected Result	- Insert fails with foreign key constraint error
Status	✅ Pass

TC-DB-007: NOT NULL Constraint

Test Case ID	TC-DB-007
Module	Database
Test Scenario	Insert member without required field
Test Steps	1. Try to insert member without name
Expected Result	- Insert fails with NOT NULL constraint error
Status	✅ Pass

---

6. SECURITY TEST CASES

6.1 Authentication Security Tests

TC-SEC-001: Password Hashing

Test Case ID	TC-SEC-001
Module	Security
Test Scenario	Verify passwords are hashed
Test Steps	1. Register member 2. Check database 3. Verify password is hashed
Expected Result	- Password stored as bcrypt hash - Original password not visible
Status	✅ Pass

TC-SEC-002: JWT Token Expiration

Test Case ID	TC-SEC-002
Module	Security
Test Scenario	Access token expires after 15 minutes
Test Steps	1. Generate access token 2. Wait 16 minutes 3. Try to use token
Expected Result	- Token rejected as expired - 401 Unauthorized error
Status	✅ Pass

TC-SEC-003: SQL Injection Protection

Test Case ID	TC-SEC-003
Module	Security
Test Scenario	Attempt SQL injection in login
Test Steps	1. Send malicious SQL in email field 2. Verify it's treated as literal string
Expected Result	- No SQL injection possible - Prisma parameterizes queries
Status	✅ Pass

TC-SEC-004: Role-Based Access Control

Test Case ID	TC-SEC-004
Module	Security
Test Scenario	Member cannot access trainer endpoints
Test Steps	1. Login as member 2. Try to access `/api/trainer/members`
Expected Result	- 403 Forbidden error - Access denied
Status	✅ Pass

---

7. INTEGRATION TEST CASES

7.1 End-to-End Workflow Tests

TC-INT-001: Complete Member Journey

Test Case ID	TC-INT-001
Module	Integration
Test Scenario	Member registers, logs in, views data
Test Steps	1. Register new member 2. Login with credentials 3. Get profile 4. View workout plans 5. View diet plans 6. Check attendance 7. View progress
Expected Result	- All operations succeed - Data consistency maintained
Status	✅ Pass

TC-INT-002: Trainer Workflow

Test Case ID	TC-INT-002
Module	Integration

Test Case ID	TC-INT-002
Test Scenario	Trainer manages member
Test Steps	1. Trainer logs in 2. View all members 3. Assign workout plan 4. Assign diet plan 5. Record attendance 6. Update progress
Expected Result	- All operations succeed - Member can see assigned data
Status	✅ Pass

TC-INT-003: Cross-Module Data Flow

Test Case ID	TC-INT-003
Module	Integration
Test Scenario	Verify data flows between modules
Test Steps	1. Trainer assigns plan to member 2. Member logs in 3. Member views plan 4. Verify plan details match
Expected Result	- Plan assigned by trainer visible to member - Trainer name included - Timestamps accurate
Status	✅ Pass

---

8. PERFORMANCE TEST CASES

TC-PERF-001: API Response Time

Test Case ID	TC-PERF-001
Module	Performance
Test Scenario	Measure API response times

Test Case ID	TC-PERF-001
Test Steps	1. Send 100 requests to each endpoint 2. Measure response time
Expected Result	- 95% of requests < 500ms - Average < 300ms
Status	⚠️ To Be Tested

TC-PERF-002: Concurrent Users

Test Case ID	TC-PERF-002
Module	Performance
Test Scenario	Handle 100 concurrent users
Test Steps	1. Simulate 100 users 2. All perform login and data retrieval
Expected Result	- All requests succeed - No timeouts - Response times acceptable
Status	⚠️ To Be Tested

9. TEST SUMMARY

Test Statistics

Category	Total	Pass	Fail	Pending
Authentication	10	10	0	0
Member Module	9	9	0	0
Trainer Module	12	12	0	0
Database	7	7	0	0
Security	4	4	0	0
Integration	3	3	0	0
Performance	2	0	0	2
TOTAL	**47**	**45**	**0**	**2**

Pass Rate: 95.7%