

SE-01-TS12 Technical Specification - Backend for Guest Onboarding and Deferred Authorization for MoodBoss

Parameter	Value
Version	v1.0
Date	22.04.2026
Status	Technical specification. Commercial estimate, cost, deadlines and payment procedure are set out in the separate document SE-01-SM12.
Basis	Agreement No. 25042501 dated 24.04.2025; MoodBoss usability audit, the need to remove the mandatory authorization barrier at first entry.
Related documents	SE-01-SM12 - estimate and schedule; current MoodBoss backend architecture: auth, users, calculations, notifications, purchases.

1. Purpose of the Document

This technical specification defines the backend scope of work for guest onboarding and deferred authorization in MoodBoss. The document describes the target guest user model, deferred binding of email / Google / Apple, preservation of data on the same user_id, functional and non-functional requirements, responsibility boundaries and acceptance criteria.

2. Objective

The MoodBoss backend onboarding logic must be changed so that the user can start using the application without mandatory authorization on first entry.

Target result:

- the user enters the application without email and password;
- the user immediately fills in profile data;
- the backend immediately creates a user entity;
- the calendar and emotional weather are calculated and saved immediately;
- at the first stage, these data belong to a guest account;
- later, in profile settings, the user can bind the same account to email + password, Google Sign-In or Apple Sign-In;
- after binding, the entire accumulated profile, statistics, calendar, emotional weather, media and other data remain on the same user_id.

3. Important Architectural Decision

Do not use Google Analytics ID as the primary backend user identifier.

Recommended approach:

- use a proprietary stable anonymous installation identifier generated by the client application;
- use install_id / anonymous_id as the technical basis for starting a guest session;
- store GA ID / app_instance_id / user_pseudo_id only as an auxiliary analytical identifier, not as the primary account key.

Reasons:

- GA ID is not designed to act as a login or account identity;
- GA ID may be unstable and dependent on analytics infrastructure;
- binding business data to GA ID creates unnecessary risks during migrations, reinstallations and SDK changes.

4. Target Backend Model

The optimal model for the current architecture:

- create a regular backend user immediately;
- mark this user as a guest account;
- issue regular JWT tokens;
- continue working in all services through the same user_id;
- later upgrade the same account to a regular account by binding email / Google / Apple.

This model avoids breaking the current JWT-based authorization, avoids deep rewriting of calculations, users, notifications and other services, and avoids migrating calculation data between two accounts.

5. Target User Flow

Scenario	Steps	Result
First application launch	The client generates or loads a stable anonymous install identifier. The application calls the backend endpoint to start a guest session. Backend creates a guest user and returns access / refresh tokens.	The application works with a regular authorized JWT session without email / password.
Initial profile completion	The application sends profile data to the backend. Backend saves the profile on the guest user and starts the existing recalculation pipeline.	Calendar and emotional weather are available immediately within the same user_id.
Deferred account binding	In profile settings, the user selects email + password, Google or Apple. Backend verifies the selected method and binds the identity to the current user_id.	The account stops being a guest account, the history remains in place.
Subsequent logins	The user logs in through email/password, Google or Apple. Backend searches for the bound identity.	The same account and the same user_id are opened.

6. Backend Scope by Services

6.1 Users Service

Required

- extend UserModel to support guest accounts;
- add storage of anonymous installation identifier, for example install_id_hash;
- add guest account state fields: is_guest, registered_at, upgraded_at, registration_source if required;
- preserve email uniqueness, but use placeholder email for guests;
- set an unusable password for guest users until email/password binding;
- create push settings for guest users in the same way as for regular users;
- return guest-state information in /users/api/v2/my;
- preserve the current profile-saving and recalculation-start logic.

It is recommended to add a separate table of social identities UserIdentityModel:

- user_id;
- provider (google, apple);
- provider_subject;
- provider_email - optional;
- timestamps;
- unique constraint on (provider, provider_subject).

New gRPC methods or an equivalent internal service layer:

- CreateGuestUser;
- BindGuestToEmail;
- LinkSocialIdentity;
- AuthBySocialIdentity;
- GetUserByInstallId - optional for idempotency.

6.2 Auth Service

Required

- add the guest session start endpoint POST /auth/api/v1/guest/start;
- add the email binding flow for the current guest user: sending a confirmation email, confirming email, setting a password, converting the guest account into a regular account;
- add Google binding flow;
- add Apple binding flow;
- add login through Google;
- add login through Apple;
- preserve the current email/password login for regular users;
- preserve JWT issuance for the same user_id;

- ensure that Redis logic for storing active tokens is compatible with guest users.

Endpoint	Purpose
POST /auth/api/v1/guest/start	Start a guest session and issue JWT.
POST /auth/api/v1/bind/email/send	Send email confirmation message.
POST /auth/api/v1/bind/email/confirm	Confirm email and set password.
POST /auth/api/v1/bind/google	Bind Google identity to the current guest account.
POST /auth/api/v1/bind/apple	Bind Apple identity to the current guest account.
POST /auth/api/v1/google/login	Login through Google to an existing bound account.
POST /auth/api/v1/apple/login	Login through Apple to an existing bound account.

6.3 Calculations Service

Minimal changes are expected. It is necessary to verify that guest JWT works in the same way as the current JWT, because the payload contains a regular user_id. A regression check is mandatory for female calculations, male calculations, calendar, emotional weather, current day endpoints, metrics and media flows. Targeted fixes are performed only where there is an implicit expectation that the user already has an email or non-guest state.

6.4 Other Backend Services

For notifications, purchases and other dependent services, compatibility with guest users must be checked. Major architectural changes are not expected if the services already operate through user_id. Only identified compatibility issues are corrected.

7. Functional Requirements

Block	Requirement
Guest account creation	Backend creates guest user once per new install_id, returns valid access / refresh tokens, creates default push settings and allows access to protected profile endpoints.
Guest profile update	Backend accepts the first profile from a guest user, saves the profile in the existing user and profile structures, starts the current recalculation queue and makes calculations available within the same account.
Email binding	Backend allows the guest, from an authorized session, to send an email confirmation message, confirm email ownership, set a password after confirmation and convert the guest account into a regular email account. If the email is already bound to another account, a business error is returned unless merge is separately agreed.
Google / Apple binding	Backend validates the provider token backend-side, extracts a stable provider subject (sub), binds the verified identity to the current guest account, prevents one social identity from being bound to multiple accounts and supports subsequent login to the same account.
Collisions with existing accounts	In the first version, automatic merge of existing accounts is not performed. If an email / Google / Apple identity is already bound to another user, backend returns a clear business error.

8. Non-Functional Requirements

- full backward compatibility for existing email/password users;
- safe production migrations;
- automated tests for all new auth flows;
- backend-side verification for Google and Apple;
- silent reassignment of history between different user_id values is not allowed;
- no dependency on GA ID as the primary account key.

9. Out of Scope of this Technical Specification

- mobile development;
- frontend / UI in settings;
- implementation of analytics SDK;
- BI / reporting;
- engine for merging two accounts;
- support tooling for manual merge;
- subscription redesign beyond guest compatibility checks;
- texts, localization and product scenarios outside the minimum required backend part

10. Acceptance Criteria

1. A new user can start a guest session without email/password.
2. Guest receives a valid JWT and can call existing protected profile endpoints.
3. Guest can save the profile and receive calculations.
4. Calculations remain bound to the same user_id.
5. Guest can later bind email/password without losing history.
6. Guest can later bind Google without losing history.
7. Guest can later bind Apple without losing history.
8. Subsequent login through email / Google / Apple opens the same account.
9. Existing regular users continue to work without regressions.
10. Automatic account merge is not performed unless it is separately agreed.

11. Backend Work Composition

The scope of work under this technical specification includes.

- guest account model changes and migrations;
- support for anonymous install identifier;
- guest-start endpoint;
- email binding flow for guest;
- Google bind + login flow;
- Apple bind + login flow;
- social identity table;
- compatibility changes in /users/api/v2/my;
- calculations compatibility check;
- automated backend tests;
- release preparation and migration support.

12. Division into Stages

Stage	Composition
Stage 1 - Guest + deferred email binding	guest-start; guest user model; support install identifier; profile saving; calculations compatibility; email bind flow.
Stage 2 - Google + Apple bind and login	identity table; Google bind/login; Apple bind/login; collision rules and tests.

13. Main Risks

- using GA ID as primary account identity is not recommended;
- Apple email may be hidden or absent, therefore sub must be the unique key;
- automatic merge of existing accounts is a separate and more risky scope;
- if the mobile application cannot reliably store install_id, guest continuity after reinstall will be unreliable.
- Google / Apple integration may refine the estimate depending on the exact mobile contract.

14. Assumptions

- the scope includes backend only;
- the client will pass a stable anonymous install identifier;

- the client will receive Google / Apple identity tokens and pass them to the backend for verification;
- the first version does not require merging an existing regular account with a new guest account.
- the current Redis/JWT authorization model is preserved.

15. Responsible Persons and Areas of Responsibility

Party	Responsible Person	Area of Responsibility
Contractor	Vitaliy Stanislavovich Dildin	Organization of backend works, task-setting, implementation control, preparation of the result for delivery
Customer	ONERY OVERSEAS LIMITED / authorized representatives	Provision of source data, confirmation of requirements, participation in result verification and acceptance.
Testing and acceptance	Vladislav Ivanovich Zverev	Participation in guest flow verification, regression-check and confirmation of result readiness for approval.

Signatures of the Parties

Contractor		Customer	
Vistadi LLC		ONERY OVERSEAS LIMITED	
Director	 V. S. Dildin	Director:	 Boulitsidou A.

