Low Level Design Document

Introduction

This Low Level Design (LLD) document outlines the core components, data structures, and logic for **AdWise - AI Digital Marketing Campaign Builder**. AdWise is a platform leveraging generative AI to create, optimize, and analyze digital marketing campaigns, supporting secure user management, collaborative editing, and exportable reports.

---

1. System Components

Component	Technology	Key Responsibilities
Frontend UI	React	User interface, campaign builder, collaboration
Backend API	Node.js/Express	Business logic, AI orchestration, data access
AI Service Integrator	Node.js	Connects to generative AI APIs (text/image)
Database	MongoDB	Stores users, campaigns, assets, analytics
Auth Service	Node.js/JWT	User authentication, authorization
Report Generator	Node.js	Generates/export reports (PDF/CSV)
Real-time Collaboration	WebSockets	Enables multi-user editing

---

2. Class/Interface Overview

Class/Interface	Relationships	Key Methods/Attributes
`User`	Owns Campaigns, Teams	`id` , `email` , `role` , `passwordHash`
`Campaign`	Has Ads, Analytics	`id` , `name` , `ownerId` , `ads[]` , `status`
`Ad`	Belongs to Campaign	`id` , `type` , `copy` , `visual` , `channel`
`Analytics`	Linked to Campaign/Ad	`impressions` , `clicks` , `ctr` , `roi`
`Team`	Has Users, Campaigns	`id` , `name` , `userIds[]`
`AIService` (interface)	Used by Backend	`generateCopy()` , `generateVisual()`
`Report`	Generated from Analytics	`campaignId` , `format` , `data`

---

3. Data Structure Overview

Model	Fields
User	`_id` , `email` , `passwordHash` , `role` , `teamIds[]`

Campaign	`_id`, `name`, `ownerId`, `teamId`, `ads[]`, `status`, `createdAt`
Ad	`_id`, `campaignId`, `type`, `copy`, `visualUrl`, `channel`, `status`
Analytics	`_id`, `adId`, `impressions`, `clicks`, `ctr`, `roi`, `timestamp`
Team	`_id`, `name`, `userIds[]`, `campaignIds[]`
Report	`_id`, `campaignId`, `format`, `generatedAt`, `data`

---

4. Algorithms/Logic

Campaign Generation Flow (Pseudocode):

```
function createCampaign(userId, campaignInput):
    validateUser(userId)
    campaign = new Campaign(campaignInput)
    for each adSpec in campaignInput.ads:
        adCopy = AIService.generateCopy(adSpec)
        adVisual = AIService.generateVisual(adSpec)
        ad = new Ad(adCopy, adVisual, adSpec.channel)
        campaign.addAd(ad)
    save campaign to DB
    return campaign
```

Analytics Aggregation:

```
function getCampaignAnalytics(campaignId):
    ads = fetchAdsByCampaign(campaignId)
    analytics = aggregate(ads.analytics)
    return analytics
```

---

5. Error Handling

Scenario	Handling Approach
Invalid user authentication	Return 401 Unauthorized, log attempt
AI service failure/time-out	Retry, fallback to default, notify user
Data validation error	Return 400 Bad Request with error details
Database connectivity issue	Return 503 Service Unavailable, log error
Unauthorized resource access	Return 403 Forbidden
Collaboration conflict	Use last-write-wins, notify users of conflict

---

End of Document