MedNAIS Authentication System

Overview

The MedNAIS SOP Marketplace now uses a complete, production-ready authentication system with email verification.

Features

1. Registration Flow

- Users register with email, password, and name
- Password is securely hashed using bcrypt (12 rounds)
- Unique verification token is generated
- User account is created in database (unverified)
- Confirmation email is sent from `max@samplify.org`
- Email contains clickable verification link
- Account is activated when user clicks the link

2. Login Flow

- Users login with email and password
- System validates credentials against hashed password
- System checks if email is verified
- Unverified users cannot login
- Verified users receive JWT session token
- Session lasts 30 days

3. Email Verification

- Professional Russian-language emails
- Clickable verification link
- One-time verification tokens
- Success/failure messages on verification

Database Schema Changes

Added to User model:

```
password           String?   // Hashed password
verificationToken  String?   @unique // Token for email verification
```

Technical Implementation

Authentication Provider

- Uses NextAuth.js v4 with JWT strategy
- CredentialsProvider for email+password authentication

- Prisma adapter for database integration

Security Features

- Passwords hashed with bcrypt (12 rounds)
- JWT sessions with 30-day expiry
- HttpOnly cookies for session tokens
- Email verification required before login
- One-time verification tokens
- CSRF protection via NextAuth

Email Configuration

- SMTP: smtp.gmail.com:587
- From: max@samplify.org
- Uses nodemailer for email delivery
- Professional HTML email templates

API Endpoints

`/api/signup` (POST)

Registers a new user

Request:

```
{
  "email": "user@example.com",
  "password": "password123",
  "name": "John Doe",
  "role": "buyer" // or "seller"
}
```

Response:

```
{
  "message": "Пользователь создан успешно. Проверьте ваш email для подтверждения
аккаунта.",
  "user": {
    "id": "...",
    "email": "user@example.com",
    "name": "John Doe",
    "role": "buyer"
  }
}
```

`/api/auth/verify-email?token=xxx` (GET)

Verifies user email address

Success: Redirects to `/auth/signin?message=verified`
Already Verified: Redirects to `/auth/signin?message=already-verified`
Invalid Token: Returns error message

`/api/auth/[...nextauth]`

NextAuth.js endpoint for authentication

Pages

`/auth/signup`

- Registration form with email, password, name, and role
- Shows success message after registration
- Links to signin page

`/auth/signin`

- Login form with email and password
- Displays success messages from email verification
- Shows error messages for invalid credentials or unverified accounts
- Links to signup page

Testing

Test Registration:

1. Go to https://sop-marketplace-2xsu5a.abacusai.app/auth/signup
2. Fill in the form:
   - Name: Test User
   - Email: your-email@example.com
   - Password: test1234
   - Role: Buyer or Seller
3. Click “Создать аккаунт”
4. You should see a success message
5. Check your email for verification link

Test Email Verification:

1. Check email inbox for message from max@samplify.org
2. Click the verification link
3. You should be redirected to signin page with success message

Test Login:

1. Go to https://sop-marketplace-2xsu5a.abacusai.app/auth/signin
2. Enter your email and password
3. Click “Войти”
4. You should be redirected to marketplace

Test Unverified Account:

1. Try to login before verifying email
2. You should see error: “Пожалуйста, подтвердите ваш email перед входом”

Test Invalid Credentials:

1. Try to login with wrong password
2. You should see error: “Неверный email или пароль”

Error Messages (Russian)

- **Неверный email или пароль** - Invalid credentials
- **Пожалуйста, подтвердите ваш email перед входом** - Email not verified
- **Пользователь с таким email уже существует** - User already exists
- **Пароль должен содержать минимум 4 символа** - Password too short
- **Email обязателен** - Email required
- **Пароль обязателен** - Password required

Files Modified/Created

New Files:

- `app/api/auth/verify-email/route.ts` - Email verification endpoint
- `lib/email.ts` - Email sending utility with nodemailer
- `AUTHENTICATION_SYSTEM.md` - This documentation

Modified Files:

- `prisma/schema.prisma` - Added password and verificationToken fields
- `app/api/signup/route.ts` - Complete rewrite for password hashing and email verification
- `lib/auth-options.ts` - Updated to use bcrypt and check email verification
- `app/auth/signup/page.tsx` - Added password field and success message
- `app/auth/signin/page.tsx` - Improved error handling and verification messages

Environment Variables

Required in `.env`:

```
EMAIL_SERVER_HOST=smtp.gmail.com
EMAIL_SERVER_PORT=587
EMAIL_SERVER_USER=max@samplify.org
EMAIL_SERVER_PASSWORD=xbiq lcqd bgxl oaae
EMAIL_FROM=max@samplify.org
NEXTAUTH_URL=https://sop-marketplace-2xsu5a.abacusai.app
NEXTAUTH_SECRET=your-secret-key
```

Dependencies Added

```
{
  "bcryptjs": "^3.0.3",
  "@types/bcryptjs": "^3.0.0",
  "nodemailer": "^7.0.11",
  "@types/nodemailer": "^7.0.4"
}
```

Security Considerations

1. **Password Storage**: Passwords are never stored in plain text, only bcrypt hashes
2. **Verification Tokens**: One-time use tokens that are cleared after verification

3. **Session Management**: JWT tokens stored in httpOnly cookies
4. **Email Verification**: Required before account activation
5. **SMTP Security**: Uses TLS for email transmission

Troubleshooting

Email Not Received:

- Check spam/junk folder
- Verify SMTP credentials in .env
- Check email server logs in application console

Cannot Login After Verification:

- Ensure email verification link was clicked
- Check database: `emailVerified` should not be null
- Verify password is correct

Build Errors:

- Run `yarn prisma generate` after schema changes
- Run `yarn prisma db push` to update database
- Ensure all dependencies are installed

Production Checklist

- [x] Passwords hashed with bcrypt
- [x] Email verification implemented
- [x] SMTP configured for production
- [x] Russian language UI
- [x] Error handling implemented
- [x] Session management configured
- [x] Database schema updated
- [x] TypeScript compilation passes
- [x] Build succeeds without errors

Next Steps

1. Test the registration flow
2. Verify email delivery
3. Test login with verified account
4. Monitor email delivery in production
5. Consider adding password reset functionality (future enhancement)