



**TOBB ETU**  
**Economy & Technology University**  
**BIL 481**

**Deployment Plan**

**Group: EYAY**

**Project: HAYAI ET**

**EYAY**

Ali Türkücü

Esra Alparslan

Muhammed Yusuf Kartal

Yağız Can Akay

## **Contents**

1. Deployment Overview.....	3
2. Deployment Process .....	4
3. Configuration Plan.....	5
4. Document-Specific Task Matrix .....	6

## 1. Deployment Overview

This document outlines the deployment strategy for the **HAYAI ET** demo presentation. To ensure a reliable, accessible, and cost-effective demo, we have adopted a cloud-based deployment approach utilizing managed services that align with our technology stack (React, FastAPI, MongoDB) and budget constraints.

### Deployment Environment

The project is deployed to a **Public Cloud Environment** to allow access from any device (tablet/laptop) during the demo without relying on local tunneling.

- **Frontend:** Deployed on **Vercel**.
  - *Rationale:* Vercel provides native support for our React application (Create React App), offering a global CDN and automatic deployments via GitHub integration.
- **Backend:** Deployed on **Render** (Web Service).
  - *Rationale:* Render provides a stable Python environment for FastAPI with native HTTPS support. We utilize the free tier, managing resources by offloading heavy assets to Cloudinary.
- **Database:** Hosted on **MongoDB Atlas** (Cloud).
  - *Rationale:* Uses the M0 Sandbox (Free Tier) for storing user data and metadata (JSON documents).
- **Image Storage (New):** Hosted on **Cloudinary**.
  - *Rationale:* To avoid the strict 512MB storage limit of MongoDB's free tier, all user drawings and AI-generated images are stored in Cloudinary. Only the image URLs are stored in the database.
- **AI Services:** Accessed via external API calls to **OpenAI** (gpt-image-1) via the backend.

### Tools Used

- **GitHub:** Source control (Monorepo structure) and trigger for CI/CD pipelines.
- **Postman:** Used to verify API connectivity post-deployment.
- **Cloudinary SDK:** Integrated into the backend for seamless image uploading.

## 2. Deployment Process

The following step-by-step process was followed to deploy the HAYAI ET project from our single GitHub repository (Monorepo) containing both frontend and backend folders.

### Step 1: Database Setup (MongoDB Atlas)

1. Log in to the MongoDB Atlas project dashboard.
2. Navigate to **Network Access** and whitelist the IP address `0.0.0.0/0` to allow connections from Render.
3. Navigate to **Database** and click "Connect" -> "Drivers" to generate the connection string.
4. Copy the connection string (SRV URI) for the backend configuration.

### Step 2: Backend Deployment (Render)

1. Connect the GitHub repository (`hayai-et`) to the Render dashboard.
2. Select "Web Service" and configure the environment to **Python 3.11**.
3. **Root Directory Configuration:** Set to backend.
  - o *Note:* This tells Render to ignore the frontend folder and focus on the FastAPI app.
4. **Build Command:** `pip install -r requirements.txt`
5. **Start Command:** `uvicorn main:app --host 0.0.0.0 --port $PORT`
6. Input the Environment Variables (detailed in Section 3).
7. Trigger deployment.
8. **Verification:** Visit <https://hayai-et-backend.onrender.com/docs> to ensure the Swagger UI loads.

### Step 3: Frontend Deployment (Vercel)

1. Connect the GitHub repository (`hayai-et`) to Vercel.
2. **Root Directory Configuration:** Click "Edit" and select the frontend folder.
3. **Framework Preset:** Select **Create React App**.
4. Add the Environment Variable:
  - o `REACT_APP_API_BASE_URL`: Set to the Render backend URL.
5. Deploy the project.
6. **Verification:** Access the provided `vercel.app` URL and attempt a login to verify connectivity with the backend.

### 3. Configuration Plan

The configuration relies on environment variables to secure secrets and connect the services. Below is the final configuration used for the live demo.

#### Backend Configurations (Render)

These variables are set in the Render Dashboard under the "Environment" tab.

Variable Key	Description / Purpose
MONGODB_URI	The connection string for MongoDB Atlas (e.g., <code>mongodb+srv://...</code> ).
DATABASE_NAME	The specific database name (e.g., <code>hayai_et_db</code> ) to store collections.
OPENAI_API_KEY	Authentication key for generating AI images via gpt-image-1
CLOUDINARY_CLOUD_NAME	Identifies our specific Cloudinary storage bucket.
CLOUDINARY_API_KEY	Public key for authenticating image uploads.
CLOUDINARY_API_SECRET	Private secret for signing upload requests (Hidden).

#### Frontend Configurations (Vercel)

These variables are set in the Vercel Project Settings.

Variable Key	Description / Purpose
REACT_APP_API_BASE_URL	<code>https://hayai-et-backend.onrender.com/api</code> Points the frontend React app to the live Render backend.

#### 4. Document-Specific Task Matrix

Document Task	Esra	Yağız	Yusuf	Ali
Deployment Overview		✓		
Deployment Process		✓		
Configuration Plan		✓		
Formatting & Final Review	✓	✓	✓	✓